ALLA SCOPERTA DEL QL IL COMPUTER SINCLAIR

ecco l'opportunità di scoprire la potenzialità del Super BASIC, l'avanzato linguaggio di cui è corredato il QL.

di ANDREW NELSON

edizioni JCE

ALLA SCOPERTA DEL QL IL COMPUTER SINCLAIR

ecco l'opportunità di scoprire la potenzialità del Super BASIC, l'avanzato linguaggio di cui è corredato il QL

di ANDREW NELSON

Traduzione a cura di: Severino Grandi

edizioni JCE

Via dei Lavoratori, 124
CINISELLO BALSAMO (MI)

Tutti i diritti sono riservati, nessuna parte del presente libro puo' essere riprodotta, posta in sistemi di archiviazione, trasmessa in qualsiasi forma o mezzo elettronico, meccanico, di fotocopiatura, ecc., senza l'autorizzazione scritta dell'Editore.

I programmi contenuti in questo libro sono inseriti a scopo puramente dimostrativo. La loro verifica e' stata eseguita con cura, in ogni caso l'editore non risponde dei possibili errori nei listati dei programmi e del mancato funzionamento degli stessi.

Prima edizione in Gran Bretagna
INTERFACE PUBBLICATIONS
9-11 Kensingtom High Street
London W8 5nP

prima edizione in Australia
INTERFACE PUBBLICATIONS
62 Fulton Street
Clayton, Vic, 3168

Copyright © Tim Hartnell, 1984
Prima edizione Giugno 1984

Copyright per l'edizione italiana: © Edizioni JCE, 1984
Prima edizione OTTOBRE 1984

Stampato in Italia da:
Gemm Grafica S.r.l.
Via Magretti-Paderno Dugnano (MI)

PRESENTAZIONE

Il contenuto dell'opera originale era basato su edizioni provvisorie sia del QL che della relativa documentazione della Sinclair Research, la quale aveva dichiarato le sue intenzioni di sviluppare ulteriormente il sistema.

In questo mio lavoro pero', mi sono valso della documentazione ufficiale fornita con la prima versione in commercio del QL.

Pertanto sono state eliminate, per quanto possibile, le inesatte definizioni delle implementazioni di alcuni comandi, riscontrate nell'originale, e sono stati inseriti dettagli e chiarimenti su alcune parti ritenute di maggior rilievo.

In alcuni casi ho aggiunto ex novo le definizioni di comandi e procedure che non erano presenti nel QL alla stesura dell'opera. In altri casi, come ad esempio per il controllo del suono, ho preferito non addentrarmi, dato che la complessita' e la specificita' dell'argomento esulavano dagli scopi di questo libro.

Infine, fra le appendici ho provveduto a delineare brevemente i quattro programmi applicativi della Psion che peraltro occupano una parte notevole dei manuali del calcolatore.

Severino Grandi
Milano, 1984

PREFAZIONE

Andrew Nelson ha prodotto una notevole mole di lavoro negli ultimi 18 mesi. Libri di giochi per il TI 99/4A e il VIC 20 (pubblicati dalla Virgin Books), per lo Spectrum (Addison & Wesley) e libri di programmi di avventure (Interface Pubblications).

Per quanto abbiano avuto successo, secondo me, sono sicuramente meno importanti di questo lavoro sul QL Sinclair.

Il SuperBASIC di cui e' dotato il QL e' un linguaggio che presenta incredibili capacita' e soprattutto offre al programmatore un enorme potenzialita'. Il SuperBASIC e' un linguaggio all'avanguardia almeno quanto lo e' il QL fra i calcolatori.

Nelson ci offre l'opportunita' di scoprire come il QL possa essere programmato inizialmente come se fosse il fratello maggiore dello Spectrum, e come in seguito si possano gradualmente incorporare nei programmi le nuove possibilita' aperte dal SuperBASIC.

Tim Hartnell,
Londra, 1984

NOTA:
Tim Hartnell ha scritto oltre 50 libri sui microcomputer. Fra i sui ultimi lavori trovano posto "Exploring Artificial Intelligence on your Microcomputer" e "Tim Hartnell's QL Handbook".

<<<< S O M M A R I O >>>>

Capitolo 1 - Collegamenti
Alloggiamenti espansioni
Collegamenti a monitor

Capitolo 2 - Primi passi con il QL
Tasti principali
Modalita' grafiche
Orologio in tempo reale
Colori
Parole-chiave

Capitolo 3 - Fondamenti di SuperBASIC
Quattro principi
Sostituzioni
Forzatura
Azione immediata
Numeri, numeri e ancora numeri
Editor di schermo
Esecuzione dei programmi

Capitolo 4 - Identificatori
Identificatori validi

Capitolo 5 - Forzatura
La "e" commerciale (&)
Concatenazione

Capitolo 6 - Operatori
Operatori disponibili
Ordine di precedenze
Dentro le espressioni

Capitolo 7 - Vettori e stringhe
DIMension
Vettori multidimensionali
Vettori literals
Vettori stringa

CHR$ e CODE
Il set dei caratteri
Slicing
Sostituzione di vettori
Slicing di vettori numerici
LEN

Capitolo 8 - Programmazione strutturata
SuperBASIC
Strutture
Compatibilita'
GOTO/GOSUB
ON GOTO/ON GOSUB
Ridondanza
La frase di IF
THEN diventa opzionale
Nidificazioni
Ripetizioni e cicli
Selezioni
Come sostituire IF/THEN

Capitolo 9 - Funzioni e procedure
Funzioni
Variabili LOCAL
Procedure

Capitolo 10 - Grafica
Dizionario
Colori
Controllo colori
Stipples
Modi grafici
PRINT
Separatori
AT
INK e PAPER
CLS
BORDER
FLASH
INVERSE
OVER e STRIP

UNDER
PAN e SCROLL
CSIZE

Capitolo 11 - Sistemi di coordinate
Sistema grafico
SCALE
POINT e LINE
ARC e CIRCLE
FILL
Sistema a pixel
Finestre
CURSOR

Capitolo 12 - Trattamento dei dati
READ/DATA
RESTORE

Capitolo 13 - Estensione del vocabolario
Numeri casuali
Altri comandi
CLEAR e NEW
PAUSE
REM
LIST
PEEK e POKE
INPUT
INKEY$
La tartaruga grafica

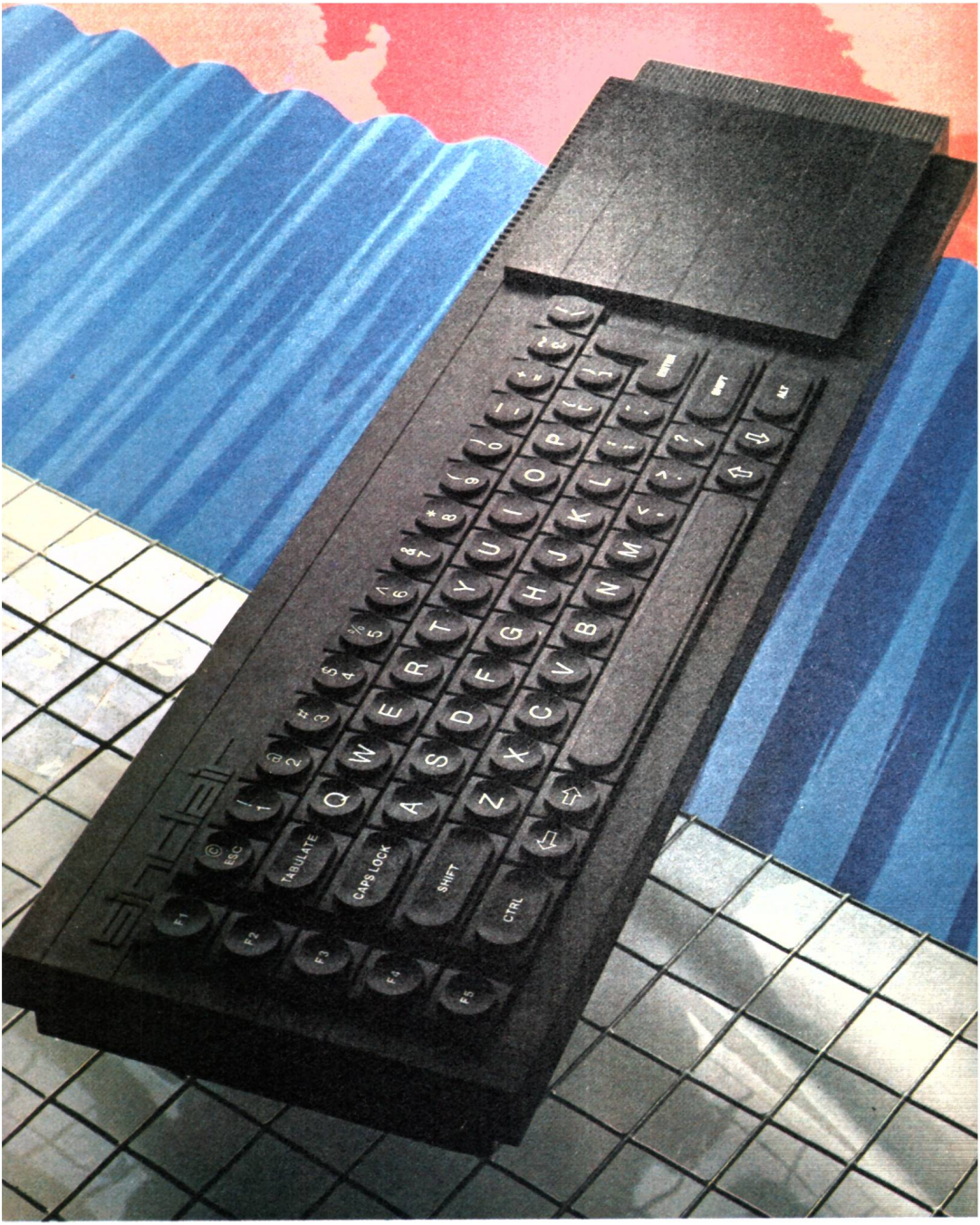
Capitolo 14 - QDOS e canali
Le mappe della memoria
Canali
Dispositivi di I/O
Controllo operazioni di I/O

Appendice A - Microdrive

Appendice B - Funzioni matematiche

Appendice C - Software professionale
QL ABACUS
QL ARCHIVE
QL BASEL
QL QUILL

Appendice D - Parole chiave del SuperBASIC



Il Sinclair QL unisce alla potenza una elegante estetica.

CAPITOLO 1

COLLEGAMENTI

Il QL viene fornito con il manuale d'uso, un alimentatore, un cavo per il televisore, uno per la rete locale (network), due astucci di plastica con le microcassette (uno con quattro micronastri in bianco, l'altro con i programmi applicativi Psion) ed infine dei piedini di plastica da mettere sotto il QL in modo da variare l'angolazione della tastiera. Con questi ultimi inseriti, l'uso del QL risultera' certamente piu' comodo. Nascosti sul retro del calcolatore vi sono degli alloggiamenti e dei connettori, attraverso i quali si potranno collegare cartucce ROM, stampanti, joystick e altro. Se si gira il QL per osservare meglio questi connettori, si vedranno nel seguente ordine (da sinistra a destra guardando la macchina da dietro):

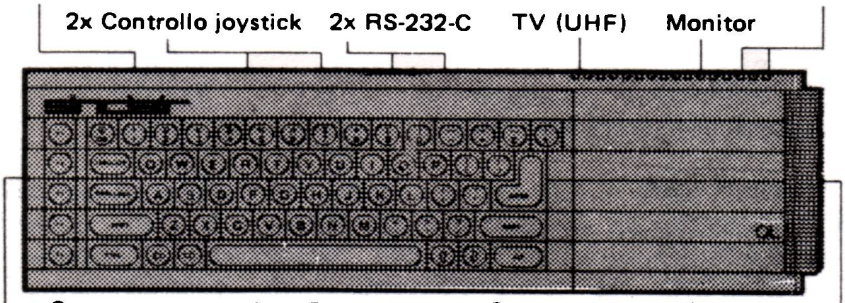
- Collegamento alla rete locale
- Collegamento alla rete locale
- Alimentazione
- Uscita monitor RGB
- Uscita TV UHF
- Prima porta seriale (RS232)
- Seconda porta seriale (RS232)
- Primo controllo joystick
- Secondo controllo joystick
- Connettore per cartucce ROM

Sulla destra, sempre dal retro, trova posto un alloggiamento per un'espansione in grado di fornire al QL un ulteriore mezzo megabyte di memoria (500.000 caratteri!) oppure per altre espansioni in progettazione. Le prese per il collegamento alla rete locale (network) consentono di connettere fra loro fino a 64 QL e Spectrum in un sistema che viene chiamato QLAN dalla Sinclair. I calcolatori componenti la rete possono "comunicare" fra loro alla

velocita' di quasi 100K baud (il baud e' l'unita' di misura della valocita' di trasmissione delle informazioni, per rendersi conto di quanto questa sia elevata, si pensi che lo Spectrum carica i programmi da cassetta a 1,5K baud). Le informazioni possono essere inviate,attraverso la rete locale, ad ogni computer che si trova "in ascolto".

Nella parte anteriore, sulla destra, vi sono gli alloggiamenti per i Microdrive. Si faccia attenzione a non inserire mai i microdrive prima che il QL sia stato acceso. Cosi' facendo si corre il rischio di alterare i dati registrati sul microdrive. Quando uno dei due microdrive sta girando e il QL sta leggendo da esso, si acccende la spia posta a sinistra del microdrive; quando questa e' accesa non si deve assolutamente provare a togliere la cartuccia, poiche' si potrebbero procurare danni sia al microdrive che al calcolatore.

La porta RGB serve al collegamento con un monitor RGB. Dopo aver sempre usato il proprio televisore, quando ci si collega ad un monitor RGB, si avra' una graditissima sorpresa: l'immagine risultera' estremamente limpida.



COLLEGAMENTO AL MONITOR

Il cavo di collegamento al monitor a colori deve avere uno spinotto DIN a otto pin da inserire nella presa RGB sul retro del QL, e una spina adatta al monitor all'altra estremita'. I monitor monocromatici, invece, possono essere allacciati al calcolatore sia con uno spinotto a otto pin (come per il colore), sia con uno a tre. Il monitor monocromatico deve inoltre avere un ingresso composito senza inversione a 75 ohm 1V pk-pk, caratteristiche che hanno quasi tutti i monitor in commercio.

Segue lo schema di collegamento degli spinotti:

PIN	FUNZIONE	SEGNALE		COL.FILO
1	VIDEO	video monocromatico composito	(3)	marrone
2	GND	massa		verde
3	-	-		arancio
4	VSYNC	sincronismo verticale	(1)	blu
5	CSYNC	sincronismo composito	(2)	giallo
6	RED	rosso	(1)	bianco
7	GREEN	verde	(1)	rosso
8	BLUE	blu	(1)	viola

(1) compatibile 0-5V TTL (attivo alto)
(2) compatibile 0-5V TTL (attivo basso)
(3) 1V pk-pk a 75 ohm

CAPITOLO 2

PRIMI PASSI CON IL QL

All'accensione del QL, si vedra' lo schermo riempirsi di strisce verticali rosse e verdi, che vengono sostituite quasi istantaneamente da strani segni, una specie di "spazzatura" da alta risoluzione. In tutto questo periodo il QL esegue automaticamente un programma di controllo interno, si accerta cioe' che tutto funzioni perfettamente. Al termine, lo schermo diventa nero e, nella parte inferiore, appare un riquadro con le scritte:

F1 for monitor
F2 for television

1983 Sinclair Research Ltd.

Premendo quindi il tasto di funzione uno (F1), si comunica al calcolatore che alla presa indicata con RBG e' collegato un monitor. Il video allora si cancella per lasciare posto a un'area di forma rettangolare che copre circa i due terzi dell'intero schermo. La meta' di sinistra di quest'area e' bianca, mentre quella di destra e' rossa. Se invece di F1 si preme F2, avvertendo cosi' il QL che alla presa UHF e' collegato un televisore, lo schermo diviene interamente bianco.

I TASTI PIU' IMPORTANTI

Vi sono quattro tasti con i quali si deve prendere confidenza quando si inizia a programmare sul QL.

Il tasto di ENTER, che e' quello a forma di "L" rovesciata, e si trova nella terza fila dal basso, a destra in prossimita' della zona dei Microdrive. Questo tasto si usa quando si vuole ordinare al QL di fare qualche cosa. Ad esempio, se e' stato immesso un comando diretto del tipo PRINT 6+7, si preme ENTER per dire al QL di procedere ad eseguire quanto gli e' stato ordinato (in questo caso, di scrivere il numero 13).

ENTER va usato anche dopo comandi quali RUN (per far partire l'esecuzione di un programma), oppure NEW (per cancellare dalla memoria del calcolatore il programma che vi si trovava). IN queste situazioni, con ENTER si richiede, come gia' detto, l'esecuzione del relativo comando.

Vi e' una terza occasione in cui si usa questo tasto: quando si immette un programma. Al termine della battitura di una linea di programma, che si vedra' via via apparire nella zona nera al disotto del rettangolo illuminato, si usa il tasto di ENTER per fare accettare la linea al QL e per fargliela aggiungere al programma.

Nel QL vi sono due tasti di scambio tastiera (SHIFT): uno a destra sotto ENTER, l'altro a sinistra sopra il tasto CTRL, nella seconda fila dal basso. Premendo uno di questi tasti (sono del tutto equivalenti fra loro), si ottengono le lettere maiuscole oppure i caratteri incisi nella parte superiore dei tasti, come ad esempio il simbolo di $ sopra il 4.

Per evitare di dover tenere premuto il tasto di SHIFT, quando si devono scrivere piu' maiuscole consecutive, si puo' bloccare la tastiera sulle maiuscole con il tasto CAPSLOCK (posto sopra lo SHIFT di sinistra). In questo caso, tutte le lettere battute alla tastiera saranno maiuscole, a meno che non si prema un tasto SHIFT, che le rende minuscole. Per disinserire lo scambio di tastiera si deve semplicemente premere di nuovo CAPSLOCK.

Se si desidera cancellare un carattere appena battuto, si devono premere contemporaneamente il tasto di CTRL

(abbreviazione di CONTROL), che si trova sotto lo SHIFT di sinistra, insieme al tasto con la "freccia all'indietro", che sta alla sinistra della barra spaziatrice. Il tasto di CTRL, come si vedra' nel seguito, viene spesso usato con il tasto di ALT o con uno alfabetico, per ottenere particolari azioni dal calcolatore.

MODALITA' GRAFICHE

Le modalita' o modi grafici verranno trattati in dettaglio piu' avanti nel libro. Comunque, tanto per fare una breve introduzione all'argomento, il QL dispone di due modalita': MODE 256, che corrisponde alla "bassa" risoluzione con 256 pixels (ovvero picture elements o punti-disegno) orizzontali per 256 verticali, e MODE 512, che offre una risoluzione doppia della precedente (512X256).

All'accensione, se si preme il tasto F2 per indicare il collegamento al televisore, si attiva automaticamente MODE 256, e in basso sullo schermo appare il cursore costituito da un quadratino rosso porpora lampeggiante. Se viceversa si comunica al QL la presenza del monitor, allora viene attivata l'alta risoluzione ed il cursore appare rettangolare rosso e lampeggiante.

COLORI

In MODE 256 si dispone di otto colori diversi (nero, blu, rosso, viola, verde, azzurro, giallo e bianco), mentre nell'alta risoluzione sono disponibili solo quattro di essi (nero, rosso, verde e bianco). Per questo motivo le modalita' si chiamano anche MODE 8 e MODE 4 rispettivamente.

Usando pero' le diverse combinazioni di retini ("stipples") in cui e' suddiviso ogni pixel, si possono creare molti altri colori. Quattro sono i tipi di stipple (a barra verticale, a barra orizzontale, a maglia larga e a

scacchiera), con i quali si ottiene un enorme ventaglio di colori.

Con un televisore o un monitor in bianco e nero, si possono distinguere otto tonalita' di grigio nella modalita' 256X256 e quattro nella 512X256.

Il numero di caratteri rappresentabili sullo schermo dipende dalle scelte effettuate sui pixel. In assenza di scelte, con il monitor, si hanno 25 righe di 80 caratteri, mentre con il televisore, il formato varia da 40 a 60 colonne, in funzione del software utilizzato.

OROLOGIO

Il QL possiede un dispositivo che pochi altri calcolatori hanno in dotazione: un orologio in tempo reale, che viene attivato all'accensione del sistema.

Vi sono ben cinque comandi dedicati al funzionamento dell'orologio, come si puo' vedere dallo schema che segue:

COMANDO	FUNZIONE
ADATE	regola l'orologio
DATE	fornisce la data in forma di numero reale
DATE$	fornisce la data in forma di stringa
DAY$	fornisce il giorno della settimana
SDATE	aziona l'orologio

Il comando di DATE$ e' utilizzato per accedere all'orologio (come in PRINT DATE$). Con questo comando, si ottiene un calendario-orologio completo nel formato standard ISO come il seguente:

1984 JUL 08 11:20:08

PAROLE-CHIAVE

Il SuperBASIC ha delle parole-chiave riservate che si possono all'incirca dividere in tre gruppi: procedure, funzioni e comandi. Ad esse se ne possono aggiungere infinite altre, semplicemente definendole come procedure.

Le procedure gia' disponibili sono:

ADATE, SDATE, ARC, LINE, LINE_R, POINT, POINT_R, SCALE, BEEP, EDIT, AUTO, RENUM, DLINE, CALL, INPUT, PAUSE, PRINT, BAUD, CLOSE, COPY, COPY_N, DELETE, DIR, FORMAT, OPEN, OPEN_IN OPEN_NEW, POKE, POKE_W, POKE_L, RANDOMIZE, CLEAR, NEW, CONTINUE, EXEC, EXEC_N, LBYTES, LIST, LOAD, LRUN, MERGE, MRUN, NEW, RETRY, RUN, SAVE, SBYTES, SEXEC, STOP, READ, RESTORE, DIM, LOCAL, NET, MODE, AT, BLOCK, FILL, CIRCLE, PAPER, BORDER, INK, CLS, CSIZE, CURSOR, FLASH, OVER, PAN, RECOL, SCROLL, STRIP, PENUP, PENDOWN, TURN, TURN TO, UNDER e WINDOW.

Le funzioni sono:

DATE$, DATE, DAY$, DIMN, DEG, RAD, FILL$, BEEPING, INKEY$, KEYROW, ABS, ACOT, ATAN, COS, COT, EXP, INT, LN, LOG10, SIN, SQRT, TAN, PI, DIV, MOD, INSTR, CHR$, LEN, CODE, PEEK, PEEK_W, PEEK_L e RND.

I comandi sono:

LET, GOTO, GOSUB, ON GOTO, ON GOSUB, FOR, REPeat, SELect ON, IF THEN ELSE, DEFine PROCedure, DEFine FuNction, REMark, RETurn, DATA, END FOR, END IF, END REPeat, END DEFine, END SELect, REMAINDER, EXIT e NEXT.

Si noti che quando si immettono dei comandi, non e' necessario battere la parola (o le parole) per intero, basta immettere la parte indicata in maiuscole (come REP per REPeat), al resto pensa il QL.

CAPITOLO 3

ELEMENTI DI SUPERBASIC

Il SuperBASIC e' un linguaggio di programmazione sviluppato dalla Sinclair Research. Esso puo' apparire a prima vista semplicemente una versione "maggiorata" dei BASIC esistenti, cosi' come suggerisce il nome. Sicuramente, ha preso un po' dal BASIC dello Spectrum (ed e' possibile, volendo, programmare il QL piu' o meno come uno Spectrum), pero' il SuperBASIC e' molto ma molto di piu'.

In effetti il SuperBASIC, si discosta talmente dal BASIC (quando e' usato propriamente) che alla Sinclair sostengono che era loro seria intenzione chiamarlo con un altro nome, senonche' la gente avrebbe rifiutato la macchina pensando di dover apprendere un nuovo linguaggio per programmarla.

I quattro componenti fondamentali del SuperBASIC sono:

- gli identificatori
- la capacita' di raggruppamento e manipolazione di elementi correlati
- la forzatura
- lo slicing ovvero la manipolazione delle stringhe

Per favorire la trasposizione dagli altri BASIC al SuperBASIC, sono presenti molte delle parole-chiave piu' note, che non sarebbero state assolutamente necessarie. Queste "meno che essenziali" parole, lasciate per facilitare la traduzione nel BASIC del QL, permettono la trascrizione di programmi scritti per altre macchine con minime modifiche. Comunque, per ottenere il massimo dal calcolatore e per sfruttare appieno la potenza del suo linguaggio, e' necessario abituarsi a sostituire le istruzioni superate con quelle piu'appropriate del SuperBASIC.

GOTO puo' essere rimpiazzato da IF, REPeat e simili, e GOSUB puo' essere rimpiazzato da una richiamo a una procedura (definita da DEFine PROCedure). ON...GOTO e ON...GOSUB (che non erano presenti nei precedenti calcolatori Sinclair) possono essere sostituiti in SuperBASIC con SELect.

Molti comandi, con i quali tutti hanno confidenza, svolgono in SuperBASIC esattamente le stesse funzioni che negli altri BASIC. Fra questi si trovano PEEK e POKE, LIST, READ e DATA. Altri ancora, come REMark e SQRT, prendono il posto degli analoghi e piu' comuni REM e SQR.

Vi sono poi alcuni comandi che non funzionano affatto, come ad esempio LLIST, che e' usato in molti calcolatori per ottenere un listato su una stampante. Questo compito viene svolto dal QL semplicemente con LIST, dopo che sara' stato aperto un canale alla stampante. In tal modo, eseguire un LIST su stampante sara' ugualmente quanto lo e' normalmente su video.

Dunque si scoprira' che e' possibile ottenere dal QL tutto cio' che si ottiene in BASIC dagli altri calcolatori (anzi, molto di piu'), anche se qualche volta si dovra' andare alla ricerca della combinazione di comandi necessaria. Oltre a LLIST, altri comandi piuttosto comuni, e non presenti nel QL sono: LPRINT, VAL, STR$, IN, OUT, ON ERROR e IN INPUT.

Cosi' come si ottengono i listati su stampante usando LIST # (con il numero di periferica), allo stesso modo con PRINT # si ottiene l'effetto di LPRINT.

VAL e STR$ non sono fornite semplicemente perche' non sono necessarie, come si vedra' quando si trattera' la "forzatura", un caratteristico ed interessante dispositivo del QL. Come certamente e' noto, VAL trasforma in un numero una stringa contenente un espressione numerica (cioe' trasforma la stringa "1" in 1, in modo che possa essere utilizzata come un numero). STR$ esegue l'operazione inversa, trasformando un numero in stringa (il numero 456 nella stringa "456"). Nel QL cio' avviene automaticamente,

quando e' necessario, cioe' l'elemento viene "forzato" ad assumere la forma richiesta, quindi VAL e STR$ non servono.

IN e OUT, comandi presenti nello Spectrum (ed in altri calcolatori), non sono applicabili al Motorola 68008, il microprocessore principale del QL.

Per quanto riguarda ON ERROR e IN INPUT, e' possibile simularli definendo delle semplici procedure.

AZIONE IMMEDIATA

Come accennato precedentemente, una frase in SuperBASIC contiene le disposizioni per il QL, affinche' faccia qualcosa immediatamente o piu' tardi quando il programma che contiene tali disposizioni viene mandato in esecuzione. Se la frase inizia con un numero di linea, il calcolatore capisce che essa fa parte di un programma e quindi non la esegue fino a che il programma non e' stato avviato.

Se invece la frase non inizia con un numero di linea, il QL la interpreta come un comando e la esegue non appena premuto il tasto ENTER.

Quindi se si batte PRINT "CIAO" e si preme il tasto ENTER, la parola CIAO compare sullo schermo. Se, d'altra parte, si batte 10 PRINT "CIAO" e poi ENTER, la linea viene copiata nella parte alta dello schermo, per mostrare che e' stata accettata come linea di un programma. Per avere la scritta CIAO sullo schermo, occorre ora battere RUN e di seguito ENTER.

Anche una frase diretta (designata per una immediata esecuzione), cosi' come una linea di programma, puo' essere composta da piu' istruzioni. Per unire fra loro le istruzioni componenti, si usa il simbolo di due punti (:). Dunque, PRINT "CIAO": PRINT "ARRIVEDERCI" puo' essere immesso come un unico comando diretto. Dopo aver premuto ENTER, sullo schermo compaiono le due parole CIAO e ARRIVEDERCI.

NUMERI, NUMERI E ANCORA NUMERI

Coloro che hanno incominciato ad occuparsi di calcolatori con lo ZX80 ricorderanno che il numero piu' grande che si poteva manipolare era solo 32767 e il piu' piccolo -32767, o -32766 in alcuni casi. A paragone il QL e' dotato di un intervallo di numeri quanto meno stupefacente. In verita', i limiti per gli interi (numeri interi) rimangono da -32767 a +32767 e le variabili destinate a contenerli si distinguono dalle altre dall'avere il nome seguito dal simbolo di percentuale (come ad esempio VARITN%). Pero' i numeri in virgola mobile hanno un intervallo di variabilita' che supera i confini della Galassia!

Il range dei numeri in virgola mobile, infatti, si estende da -10 elevato a -615 verso destra fino a 10 elevato a 615. Cio' significa che si possono trattare numeri estremamente piccoli in valore assoluto (piu' o meno 10 elevato a -615) e numeri enormemente grandi (piu' o meno 10 alla 615-esima).

Le variabili stringa possono contenere fino a 32768 caratteri ciascuna (le stringhe verranno trattate in dettaglio un po' piu' avanti).

In tutti i precedenti modelli Sinclair i numeri di linea, che precedono le istruzioni in un programma, potevano andare da 1 (ovvero 0 per mezzo di una POKE) fino a 9999. Ebbene il range per il QL e' da 1 a 32767.

EDITOR DI SCHERMO

Il comando EDIT viene usato, ovviamente, quando si vuole effettuare l'editing di una linea di programma, cioe' quando si vuole modificare o correggere una linea. Per richiamare l'editor si deve battere un comando nella seguente forma:

EDIT nnn (dove nnn e' il numero di linea)

Se il numero di linea viene omesso, il calcolatore predispone per le modifiche la linea di programma che ha il numero piu' basso, ovvero quella iniziale. Quando una linea e' in corso di modifica, viene evidenziata ad alta intensita' luminosa per indicare che la procedura interna del QL per la verifica della sintassi non e' ancora stata eseguita su quella linea.

Una volta premuto ENTER, per segnalare la fine delle operazioni sulla linea, il QL ne verifica la sintassi. Se la linea e' senza errori, viene immessa nel programma, sostituendo la versione precedente (prima di EDIT). Il QL e' un direttore dei lavori molto severo e non rilascia la linea fino a che non ha superato l'esame della sintassi.

Se si desidera lavorare su una linea che non rientra completamente nello schermo, muovendo il cursore lungo di essa, si ottiene uno scorrimento laterale automatico, in modo che la parte interessata sia sempre visibile.

Con EDIT si puo' lavorare in due modi: inserimento e modifica (con sovrapposizione). Per passare da una modalita' all'altra basta premere il tasto di ALT. Nel modo "inserimento", tutto cio' che si batte alla tastiera viene, appunto, inserito dopo il cursore, mentre il resto della linea si sposta verso destra per far posto al nuovo testo.

Una versione pre-EDIT sia ad esempio:

```
190 M=INT(S/Q):PRINT M
```

ora, se si vuole inserire qualche cosa dopo la prima istruzione, si deve portare il cursore subito dopo i due punti e quindi si deve introdurre il nuovo testo, ad esempio

```
B=5*M:
```

la linea apparira', dunque, come segue:

```
190 M=INT(S/Q):B=5*M:PRINT M
```

e, come sara' stato notato, il resto della frase dopo i due punti scorre di un posto verso destra ad ogni carattere inserito.

Nella modalita' di "modifica" (con sovrapposizione), la linea sarebbe apparsa come segue:

```
190 N=INT(S/Q):B=5XN N
```

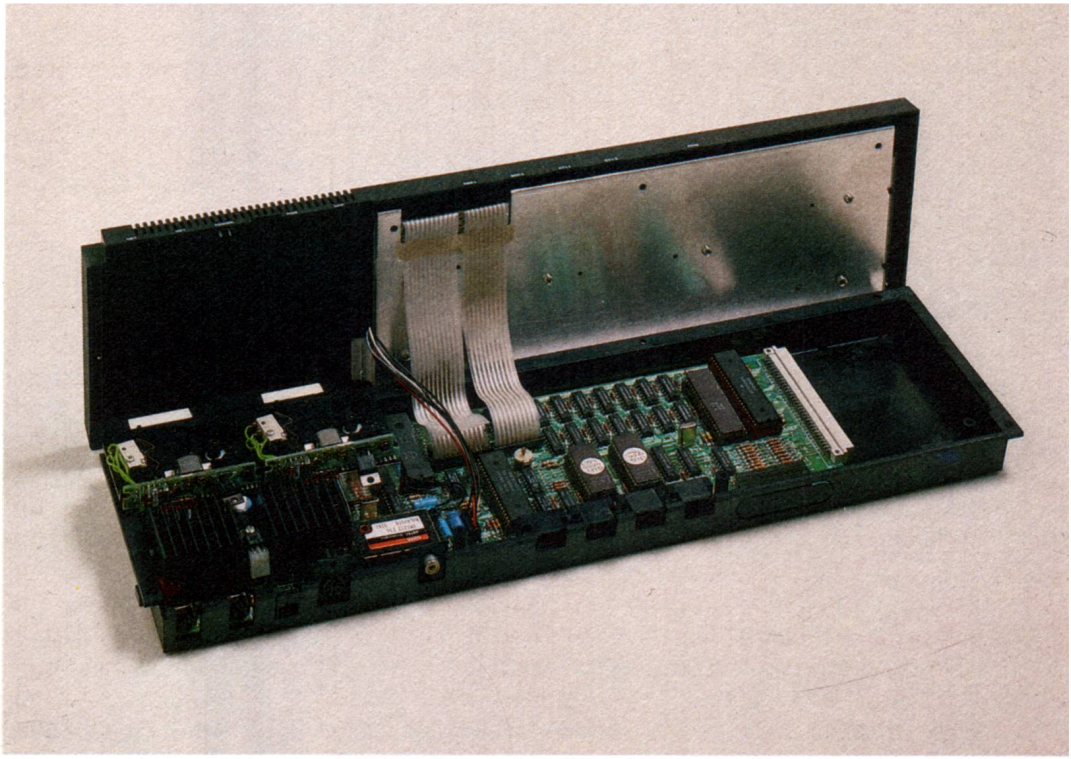
In questo, caso si dovrebbero eliminare lo spazio e la N alla fine della linea, con il tasto CTRL insieme con quello con la freccia verso destra.

Se si sta lavorando in EDIT su una linea, e si cambia idea, si puo' ripristinare la linea originaria inalterata semplicemente premendo il tasto ESC. Per uscire dalla fase di editing, al termine del lavoro su una linea, si deve premere ENTER.

ESECUZIONE DEI PROGRAMMI

Cosi'come lo Spectrum, anche il QL elimina automaticamente tutte le variabili (sia numeriche che stringa) quando viene usato RUN (e ENTER). Le variabili con i loro valori possono essere salvate facendo partire il programma con il comando GOTO 1 (sia che esista o no una linea 1 nel programma). A differenza dello Spectrum pero', l'esecuzione del programma non viene interrotta dalla richiesta di stampare il valore di una variabile non ancora definita, ma procede stampando un asterisco (X) al posto del valore. Viene segnalato l'errore, solo se si tenta si utilizzare quella variabile in espressioni. si desidera che l'esecuzione del programma inizi da una particolare linea, si puo' usare GOTO n (dove n e' il numero di tale linea), se si vogliono salvare le variabili, altrimenti si usa RUN n per cancellarle. In ogni caso l'esecuzione parte dal numero di linea specificato (oppure dalla linea con il numero successivo, se non quello non esiste nel programma).

Se l'esecuzione del programma viene interrotta per qualche motivo (ad esempio per controllare il valore di una variabile), essa puo' essere ripresa dal punto di interruzione con il comando di CONTINUE.



L'interno della macchina da un senso d'ordine grazie
ai collegamenti della tastiera eseguiti con bandella flessibile.

CAPITOLO 4

IDENTIFICATORI

Un identificatore e' un nome che si puo' assegnare ad una variabile o ad una procedura e puo' essere lungo fino a 255 caratteri. Esso deve sempre iniziare con una lettera (o con una "e commerciale" quando viene assunto come identificatore di sistema, ma questo particolare caso non viene trattato in questa sede). Il carattere iniziale puo' essere seguito da una qualunque combinazione di numeri e lettere, e - cosa molto importante - di trattini di sottolinea (_). L'identificatore viene riconosciuto come stringa, se termina con il simbolo di dollaro ($), e come variabile intera se termina con il simbolo di percentuale (%).

Il trattino di sottolinea e' particolarmente utile per creare identificatori composti di piu' parole poiche' non sono ammessi spazi intermedi. Per esempio sono identificatori validi i seguenti:

```
risultato
risposta
il_risultato_finale
indicatore_di_fine_gioco
effetto_2a
```

Si deve fare molta attenzione a non confondere il carattere di sottolinea (_) con il meno (-). Questo errore e' reso ancor piu' facile dal fatto che i due caratteri sono sullo stesso tasto, situato vicino allo zero nella primafila dall'alto. Con le minuscole, cioe' senza l'inserimento di CAPS LOCK oppure senza tener premuto SHIFT, si ottiene il meno; viceversa, con le maiuscole si ottiene il sottolinea.

Si noti che il QL (cosi' come lo Spectrum) non distingue le

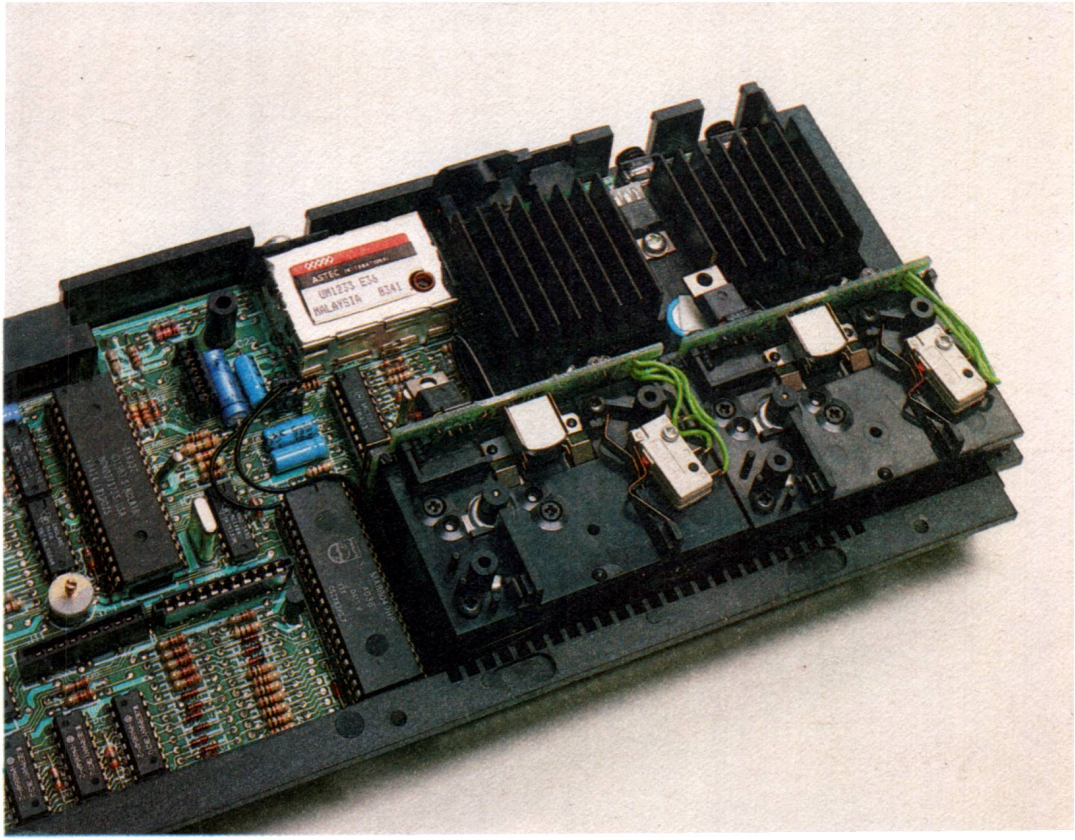
lettere maiuscole dalle minuscole negli identificatori. Cio' significa che, per il QL, i seguenti sono del tutto identici:

risposta_3e
RISPOSTA_3E
Risposta_3e
Risposta_3E
rISposTA_3e

L'identificatore e' un elemento fondamentale del SuperBASIC, lo si scoprira' man mano che ci si abituera' a lavorare con questo linguaggio; non vi sara' programma che ne sara' privo. Quando si riconoscera' il contributo alla chiarezza fornito ai programmi dall'uso di procedure (e quindi degli idetificatori) piuttosto che dei GOTO e GOSUB, si avvertira' quanto il Superbasic sia un potente strumento per la programmazione.

Si noti che, in alcuni casi, un identificatore puo' sembrare che si comporti come una variabile, senza esserlo. Un identificatore e' l'obiettivo che il QL punta su uno specifico oggetto che ha memorizzato. Nel nome dell'identificatore sono racchiuse molte altre informazioni come la natura di cio' che e' identificato (stringa,numero intero, numero in virgola mobile o procedura), il particolare valore assunto in quel momento (ad es. "Maria" , 1950 oppure 5.01, ecc.), ela sua posizione nella memoria del QL, e altro.

Quindi in SuperBASIC, quando si usa l'identificatore A collegato ad altri identificatori B, C, ecc., si e' in grado di manipolare non solo le informazioni contenute in A, ma anche quelle degli altri considerandoli come parti di A, ovvero si possono costruire procedure di procedure, funzioni di funzioni, e cosi' via.



La memoria di massa del QL sono i microdrives:
eccone la sezione interessata che può accogliere contemporaneamente due cartridge.

==========
C A P I T O L O 5
==========

FORZATURA

Un'altra particolarita' del SuperBASIC e' la cosiddetta "coercion" o forzatura, argomento di cui si tratta in questo capitolo. Nella stragrande maggioranza dei BASIC degli altri calcolatori, compresi quelli dello ZX81 e dello Spectrum, quando si fornisce un dato di tipo errato, si mette in crisi il sistema.

Si consideri, ad esempio, il seguente programma:

```
10 PRINT "IMMETTI UN NUMERO"
20 INPUT A
30 PRINT "IMMETTINE UN ALTRO"
40 INPUT B$
50 PRINT "LA SOMMA E' ";A + B$
```

Nella maggior parte dei calcolatori, il programma, nel corso dell'esecuzione, si blocchera' all'ultima linea, in quanto si e' tentato di far eseguire una somma tra un numero e una stringa.

Se, invece, lo si lancia sul QL, la somma viene eseguita senza problemi, supposto che sia stata immessa un'espressione a valore numerico alla seconda richiesta di immissione.

Per essere ancora piu' chiari, si supponga di aver premuto il tasto "2". Ebbene, nonostante che quello che e' stato immesso venga memorizzato come valore stringa "2" (e non come numero 2) n, il calcolatore lo traduce, ovvero lo forza, nel numero 2 in modo che possa essere sommato.

Il QL effettua la forzatura ovunque sia possibile. Nel seguente programma, che non funziona sugli altri

calcolatori, viene mostrato come il QL effettui una forzatura in senso inverso al precedente:

```
10 PRINT "COME TI CHIAMI ?"
20 INPUT A$: PRINT A$
30 PRINT "QUANTI ANNI HAI ?"
40 INPUT B: PRINT B
50 LET C$=A$ & B
60 PRINT C$
```

Si noti la presenza di una "e commerciale" (&) nella linea 50, invece del piu' comune segno piu' (+). Cio' serve ad avvertire il QL che si vuole lavorare con le stringhe e non con i numeri. Sempre a linea 50 la parola-chiave LET e' stata messa per chiarezza, nel QL e' diventata opzionale.

All'esecuzione del programma sul QL, sullo schermo appaiono le seguenti scritte:

```
COME TI CHIAMI ?
MARIA
QUANTI ANNI HAI ?
34
MARIA34
```

L'ultima riga e' il risultato della somma delle due stringhe. Naturalmente, in questo caso, l'eta' e' accettata e memorizzata nel QL come numero (nella variabile numerica B), e solo sucessivamente e' forzata al valore stringa, in modo da poter essere aggiunta all'altra (il nome). Per descrivere la somma di stringhe, come visto sopra, si usa il termine di "concatenazione".

La forzatura funziona in tutti i casi elencati nel seguito:

```
LET A = "345" + 96            (risultato 441)
LET A = "13" + "56" + "99"    (risultato 168)
LET A = 245 & 245             (risultato "245245")
LET A = "345" & 96            (risultato "34596")
```

Quando si fa eseguire al QL la somma di due numeri in virgola mobile, assegnando il risultato ad una variabile

dello stesso tipo (quelle cioe' che hanno il nome costituito da una combinazione di lettere e/o numeri, con una lettera in testa, del tipo A8976 oppure GRAN5 o GR77gr), non e' richiesta ne' effettuata alcuna forzatura.

Analogamente, se si assegna il risultato di una somma di numeri in virgola mobile ad una variabile intera (con il nome che inizia con una lettera e termina con il simbolo di % come A785%,B% oppure BAH%), non avviene alcuna forzatura, solo che il risultato viene memorizzato come numero intero (vengono cioe' ignorate le eventuali cifre decimali).

L'uso della "&" e del "+" deve essere curato attentamente. Si provi, infatti ad indovinare i risultati dei seguenti due programmi (dove il punto corrisponde alla nostra virgola decimale, secondo la consueta notazione internazionale):

I PROGRAMMA	II PROGRAMMA
10 A$ = "6754.65"	10 A$ = "6754.65"
20 B = 152.76	20 B = 152.76
30 C$ = A$ + B	30 C$ = A$ & B
40 PRINT C$	40 PRINT C$

Per il primo programma il risultato e' "6907.41", mentre per il secondo e' "6754.65152.76". Infatti nel primo, il QL forza la stringa "6754.65" al numero 6754.65 e, lasciando inalterato il valore numerico di B, li somma e sucessivamente converte il risultato (numerico) in un contenuto di stringa.

Nel secondo programma, il contenuto della stringa A$ rimane inalterato, mentre 152.76 viene forzato a "152.76", ed in seguito le due stringhe vengono concatenate. Il segno + o la & confermano al QL il tipo di variabile da trattare.

CAPITOLO 6

OPERATORI

Il QL dispone dei seguenti operatori:

op.	significato	tipo argomento	operazione
=	uguale	numerico stringa	uguaglianza logica confronto di tipo 2
==	equivalente	numerico stringa	quasi uguale a confronto di tipo 3
+	piu'	numerico	addizione
-	meno	numerico	sottrazione
*	per	numerico	moltiplicazione
/	diviso	numerico	divisione
<	minore	numerico stringa	minore di confronto di tipo 2
>	maggiore	numerico stringa	maggiore di confronto di tipo 2
<=	minore o uguale	numerico stringa	minore o uguale a confronto di tipo 2
>=	maggiore o uguale	numerico stringa	maggiore o uguale a confronto di tipo 2
<>	diverso	numerico stringa	diverso da confronto di tipo 3

&	e commerciale	stringa	concatenazione
&&	et	binario	AND fra bit
||	vel	binario	OR fra bit
^^	aut	binario	XOR fra bit
~~	non	binario	NOT,complemento a 1
AND	et	valore logico	congiunzione logica
OR	vel	valore logico	inclusione logica
XOR	aut	valore logico	esclusione logica
NOT	non	valore logico	negazione logica
MOD	modulo	numerico intero	resto divisione
DIV	diviso	numerico intero	divisione fra interi
^	elevato	numerico	elevazione a potenza
(^)	elevato	numerico intero	elevazione a potenza
-	meno unario	numerico	cambiamento di segno
+	piu' unario	numerico	nessuna operazione

ORDINE DI PRECEDENZA

Se nel corso di un programma il QL incontrasse il seguente frammento

```
100 A=16+8X2
110 PRINT A
```

il suo risultato sarebbe 32 (16 piu' otto volte 2), se invece la linea 100 fosse

```
100 A=(16+8)X2
```

...il QL risponderebbe 48 (la somma di 16 piu' 8 moltiplicata per 2).

La differenza consiste, ovviamente, nelle parentesi. Nella seconda versione della linea 100, le parentesi garantiscono l'esecuzione della somma prima della moltiplicazione. Nel primo caso la moltiplicazione viene eseguita prima secondo la legge delle precedenze.

Si dice "ordine di precedenza" l'ordine secondo il quale vengono eseguite le operazioni. Un'operazione ha ordine di precedenza piu' alto di un'altra quando viene eseguita prima dell'altra. Se due operazioni hanno lo stesso ordine di precedenza viene eseguita la prima da sinistra verso destra. Le parentesi possono alterare l'ordine di precedenza.

Segue l'elenco delle operazioni secondo l'ordine di precedenza sul QL, dal piu' alto al piu' basso:

- piu' e meno unari
- concatenazione di stringhe
- elevamento a potenza
- moltiplicazione, divisione, DIV e MOD
- addizione e sottrazione
- confronto logico
- NOT
- AND
- OR e XOR

Gli operatori logici (>,<,=,ecc.) possono essere inclusi nelle espressioni numeriche. In tal caso le espressioni assumono valore logico, cioe' se sono vere (come PRINT 2<3) assumono valore diverso da zero (nell'es. 1), se false (come PRINT 3<>3) si ottiene il valore zero.

CAPITOLO 7

VETTORI E STRINGHE

I vettori o schiere si usano quando si vuole costruire una serie di elementi (numerici o stringa) che possano essere individuati dalla posizione che occupano nella serie.

I vettori vengono definiti dall'istruzione DIMension. In effetti un array, questo e' il nome inglese dei vettori, non e' altro che un insieme di posizioni di memoria riservate all'interno del QL. Ogni elemento di questo vettore e' identificato dal nome del vettore e da un numero (indice) che indica la sua posizione all'interno del vettore. Per ottenere un vettore di quattro elementi numerici si dovra' scrivere la seguente linea BASIC (supposto che il vettore si chiami CONTIENE):

```
10 DIMension CONTIENE(3)
```

Se, invece, si vuole un vettore numerico di 45 elementi, si deve usare una frase del tipo:

```
10 DIMension nome_del_vettore(44)
```

Si ricorda che, sul QL, e' sufficiente scrivere la parte in maiuscolo dei comandi, cioe' in questo caso DIM. Si noti, inoltre che il numero fra parentesi e' inferiore di una unita' rispetto al numero degli elementi richiesti. Questo perche' il primo elemento del vettore verra' indicato come elemento zero.

Quando si dimensionano i vettori numerici, oltre alla loro definizione e alla riservazione dello spazio in memoria, si ottiene di assegnare automaticamente zero a tutti gli elementi del vettore.

Percio' il seguente programma:

```
10 DIMension A(3)
20 PRINT A(0)
30 PRINT A(1)
40 PRINT A(2)
50 PRINT A(3)
```

... produce

```
0
0
0
0
```

mentre se si vogliono assegnare i valori agli elementi di a(3), si deve scrivere:

```
10 DIMensionA(3)
20 A(0)=10
30 A(1)=9.97
40 A(2)=898273
50 A(3)=65.8781
```

Ora, per riferirsi all'elemento che contiene 898273 (anzi, per riferirsi a 898273), tutto quello che si deve fare e' riferirsi ad A(2). Nel programma, poi, si potra' manipolare A(2) esattamente come fosse una variabile numerica qualsiasi.

VETTORI MULTIDIMENSIONALI

I vettori trattati fino ad ora sono detti monodimensionali, poiche' i loro elementi costituiscono una sola sequenza. nel QL, cosi' come nella maggior parte degli altri calcolatori, sono ammessi vettori cosiddetti multidimensionali, cioe' che hanno piu' di una serie di elementi. Per il QL, non vi sono limiti al numero di dimensioni, se non quelli imposti dalla quantita' di memoria disponibile.

Come e' stato detto, quando si definisce un vettore con un'istruzione di DIMension, il QL riserva spazio per ogni elemento del vettore. Ebbene lo spazio di memoria viene riservato per ogni elemento sia che venga utilizzato o no, e i vettori multidimensionali consumano memoria a velocita' tremenda, anche se con il QL, prima di avere problemi di carenza di memoria, se ne dovrebbe sprecare veramente tanta.

In ogni caso, per definire un vettore multidimensionale, e' necessario aggiungere un secondo (un terzo, ecc.) numero dopo il primo nell'istruzione di DIMension, ad esempio con:

```
10 DIMension TABELLA(3,2)
```

si definisce un vettore bidimensionale di quattro per tre elementi. Tale vettore puo' essere pensato come una tabella, appunto. Anche i vettori come questo, oltre a quelli ad una dimensione, al momento della definizione hanno tutti gli elementi nulli. Ecco dunque come si presenta il vettore TABELLA dopo il dimensionamento:

```
 \  0  1  2
  \---------
0 : 0  0  0
1 : 0  0  0
2 : 0  0  0
3 : 0  0  0
```

Ogni elemento di questo vettore e' individuato da due indici, ad esempio, quello all'incrocio della terza riga e della seconda colonna e' TABELLA(2,1).

VETTORI LITERALS

Tutto quanto si e' visto sui vettori fino ad ora e' necessario alla loro definizione, se pero' si vuole "riempire" un vettore, cioe' assegnare i valori a tutti i suoi elementi, si puo' procedere in tre modi diversi. Normalmente si usano le istruzioni di LET, anche sottinteso come in A(0)=10 invece di LET A(0)=10 , oppure READ, se i

valori sono stati posti in frasi di DATA, oppure ancora INPUT, immettendo i valori da tastiera. In tutti questi modi, l'assegnazione avviene individualmente sugli elementi del vettore. Nel QL esiste una quarta forma di assegnazione "globale", nella quale ad un vettore possono essere assegnati globalmente tutti i suoi componenti, come si vedra' tra breve.

In primo luogo, per i vettori cosi' assegnati (detti "literals" in inglese), si richiede l'uso delle parentesi graffe, i cui tasti si trovano proprio sopra a sinistra di ENTER. Anzi, il numero delle coppie di parentesi graffe, contenute nelle parentesi graffe piu' esterne, indica esattamente il numero delle dimensioni del vettore.

Per il resto il funzionamento del meccanismo e' molto semplice, come un unico LET per tutti gli elementi del vettore. Ad esempio con il seguente programma:

```
10 DIMension SEMPRE(3)
20 SEMPRE= {8909, 467322.2,333,786}
30 PRINT SEMPRE(0)!SEMPRE(1)
40 PRINT SEMPRE(2)!SEMPRE(3)
```

si ottiene il sequente risultato:

```
8909 467322.2
333 786
```

Dunque la linea 20 ha riempito il vettore monodimensionale SEMPRE in tutti i suoi elementi.

Nel caso dei vettori multidimensionali con

```
10 DIMension ABITO(1,1)
20 ABITO{{6,4}{9,1}
```

si ottengono le seguenti assegnazioni:

```
ABITO(0,0)=6
ABITO(0,1)=4
ABITO(1,0)=9
ABITO(1,1)=1
```

VETTORI STRINGA

Tutti i vettori precedentemente considerati,compresi ABITO, SEMPRE e gli altri, erano vettori numerici, destinati a contenere numeri.

Nel QL non potevano mancare i vettori stringa, che pero' devono essere DIMensionati in un modo leggermente differente rispetto agli altri. Se, infatti, si utilizzasse un'istruzione del tipo

DIMension NOME$(6)

... si potrebbe pensare di aver creato un vettore stringa capace di contenere sette stringhe complete, ebbene non e' cosi'. A differenza della maggior parte dei BASIC non Sinclair, in questo caso, si definisce un vettore di caratteri singoli.

Percio' si potrebbe usare come nel seguito:

```
10 DIMension VAI$(6)
20 VAI$={"D","I","A","V","O"."L"."O"}
```

Per assegnare intere parole, e' necessario aggiungere nella frase di DIMension un ulteriore indice, che determina la lunghezza delle stringhe componenti il vettore. Ad esempio, per definire e assegnare un vettore che contiene tre stringhe di lunghezza massima pari a dieci caratteri (ovvero tre stringhe tutte lunghe dieci, poiche' il QL automaticamente riempie di spazi in bianco i caratteri inutilizzati), si procede come segue:

```
10 DIMension MAGO$(2,10)
20 MAGO$={"J.RUS","T.HARTNELL","L.NORTH"}
```

Per specificare una stringa componente di un vettore, basta omettere l'ultimo indice usato nella DIMension. Il frammento

di programma seguente chiarira' meglio il concetto:

```
10 DIMension INSOMMA$(1,6)
20 INSOMMA$={"PRONTO,"ECCOLO"}
30 PRINT INSOMMA$(0)
40 PRINT INSOMMA$(1)
```

Il risultato di questo programma e'

```
PRONTO
ECCOLO
```

... piuttosto che

```
P
E
```

... come ci si potrebbe aspettare.

CHR$ E CODE

Fra le altre parole-chiave del QL particolarmente utili alla manipolazione delle stringhe vi sono CHR$ e CODE. Queste due parole svolgono funzioni fra loro complementari. CHR$ (dove CHR e' l'abbreviazione di CHaracteR) traduce un numero nel carattere corrispondente a quel numero secondo la codifica internazionale ASCII.

Quindi PRINT CHR$(46) produce la stampa di un punto, in quanto il codice (CODE) del carattere "." e' 46. Viceversa se si desidera avere il codice ASCII di un carattere, basta ordinare al QL di stamparlo con PRINT CODE("n"), dove n va sostituito con il carattere di cui si vuole il codice.

Si noti che PRINT CODE("B") produce 66, ma anche PRINT CODE("BATTAGLIA")dara'66, poiche' la funzione CODE considera solo il primo elemento della stringa argomento.

Il programma che segue produce come risultato ancora 66:

```
10 PROVA$="BANANA"
20 PRINT CODE(PROVA$)
```

Si osservi che le virgolette non sono richieste con il nome di una stringa (STRINGA$,parola$,A$ o altro), ma sono necessarie quando ci si riferisce ad un "valore" di stringa.

IL SET DEI CARATTERI

Ed ecco i caratteri e i CODE corrispondenti:

CHR	CODE	CHR	CODE	CHR	CODE	CHR	CODE	CHR	CODE	CHR	CODE
spazio	32	0	48	@	64	P	80	`	96	p	112
!	33	1	49	A	65	Q	81	a	97	q	113
"	34	2	50	B	66	R	82	b	98	r	114
#	35	3	51	C	67	S	83	c	99	s	115
$	36	4	52	D	68	T	84	d	100	t	116
%	37	5	53	E	69	U	85	e	101	u	117
&	38	6	54	F	70	V	86	f	102	v	118
'	39	7	55	G	71	W	87	g	103	w	119
(	40	8	56	H	72	X	88	h	104	x	120
)	41	9	57	I	73	Y	89	i	105	y	121
X	42	:	58	J	74	Z	90	j	106	z	122
+	43	;	59	K	75	[	91	k	107	{	123
,	44	<	60	L	76	\	92	l	108	|	124
-	45	=	61	M	77	]	93	m	109	}	125
.	46	>	62	N	78	^	94	n	110	~	126
/	47	?	63	O	79	_	95	o	111		127

SLICING

La manipolazione delle stringhe con l'uso delle sottostringhe, secondo la tecnica dello "slicing"

costituisce un altro punto di forza del SuperBASIC. Questa particolare tecnica, come si vedra', e' ammessa anche per i vettori numerici, pero' sara' bene vedere prima come funziona per le stringhe.

Prima di considerare le sottostringhe di un vettore, si esaminino le sottostringhe di una stringa. Ad esempio il seguente programma:

```
10 A$="CASINO' DI SANREMO"
20 PRINT A$
30 PRINT A$(1)
40 PRINT A$(2 TO 6)
50 PRINT A$(12 TO)
60 PRINT A$(TO 7)
```

dara' i seguenti risultati:

```
CASINO' DI SANREMO
C
ASINO
SANREMO
CASINO'
```

La prima stampa (CASINO' DI SANREMO) non e' una sorpresa per nessuno, ad A$ e' stato assegnato il valore stringa "CASINO' DI SANREMO" con la linea 10, quindi ci si aspetta di vederla stampata al completo, come richiede la linea 20.

Ma che cosa succede alla linea 30? Sembra che si stia usando A$ come un vettore, senza averlo prima dimensionato. Lo slicing, tipico della Sinclair (presente nello XZ81, nello Spectrum e ora anche nel QL), permette di trattare le sottostringhe, ovvero parti di stringa, pensando alla stringa originaria come ad un vettore di caratteri singoli. Percio' A$(1) equivale al primo carattere della stringa A$ (si noti che, in questo caso l'indice varia da 1 e non da zero).

Si esamini, alla linea 40, A$(2 TO 6) che produce la sottostringa "ASINO". Il meccanismo dello slicing, in questo

caso rivolge la sua attenzione al primo numero dentro la parentesi, cioe' 2, ed estrae la parte di stringa che va dal carattere che ha posto 2 fino a (TO) quello di posto 6, secondo numero nelle parentesi.

Questo e' il principio di funzionamento dello "slicing".

Alla linea 50 la definizione di slicing sembra rimessa in discussione, in quanto manca il secondo numero. Se non c'e' un secondo numero, il QL semplicemente va dal primo carattere specificato fino all'effettivo termine della stringa.

Come e' facile indovinare, se non c'e' alcun numero prima di TO, il QL prendera' la stringa dall'inizio, come in A$(TO 7) alla linea 60 dell'esempio.

Si noti come gli spazi fra le parole della stessa stringa contino come un carattere ciascuno.

SLICING DI VETTORI STRINGA

Fino ad ora si e' visto come funziona lo slicing con le stringhe ad una dimensione. Estendendo la logica agli elementi di un vettore stringa, tutto il meccanismo mantiene inalterato il suo funzionamento. Per osservarlo piu' in dettaglio si esamini attentamente il programma che segue, e, prima di mandarlo in esecuzione sul QL, si provi ad indovinarne i risultati.

```
 10 DIMension PORTA$(5,8)
 20 PORTA$(3)="CHIUSA"
 30 PORTA$(5)="VOLANTE"
 40 PORTA$(0)="APERTA"
 50 PORTA$(1)="LOGICA"
 60 PORTA$(2)="DIPINTA"
 70 PRINT PORTA$(0)(1 TO 3)
 80 PRINT PORTA$(1)(TO 4)
 90 PRINT PORTA$(2)(4 TO)
100 PRINT PORTA$(3)(TO 1)
110 PRINT PORTA$(4)
120 PRINT PORTA$(5)(7 TO)
```

Si confronti ora quanto previsto con i risultati sotto riportati. Se tutto coincide, significa che i concetti dello slicing in SuperBASIC sono stati compresi a fondo.

Questi sono i risultati dell'esecuzione (sono state messe le virgolette per evidenziare anche gli spazi in bianco):

```
"APE"      - nel caso di PORTA$(0)(1 TO 3)
"LOGI"     - nel caso di PORTA$(1)(TO 4)
"INTA "    - nel caso di PORTA$(2)(4 TO)
"C"        - nel caso di PORTA$(3)(TO 1)
"        " - nel caso di PORTA$(4), che non e' assegnata
"E "       - nel caso di PORTA$(5)(7 TO)
```

Si puo' osservare che PORTA$(3)(TO 1) e' sostituibile con PORTA$(3)(1), mentre PORTA$(7 TO) lo sarebbe con PORTA$(5)(7), solo se la stringa PORTA$(5) fosse lunga solo sette caratteri.

Con lo slicing e' possibile assegnare ad una stringa una sottostringa di un'altra stringa. Ad esempio:

```
A$="ESPORTATO(3 TO 7)
```

assegna il valore "PORTA" alla variabile stringa A$.

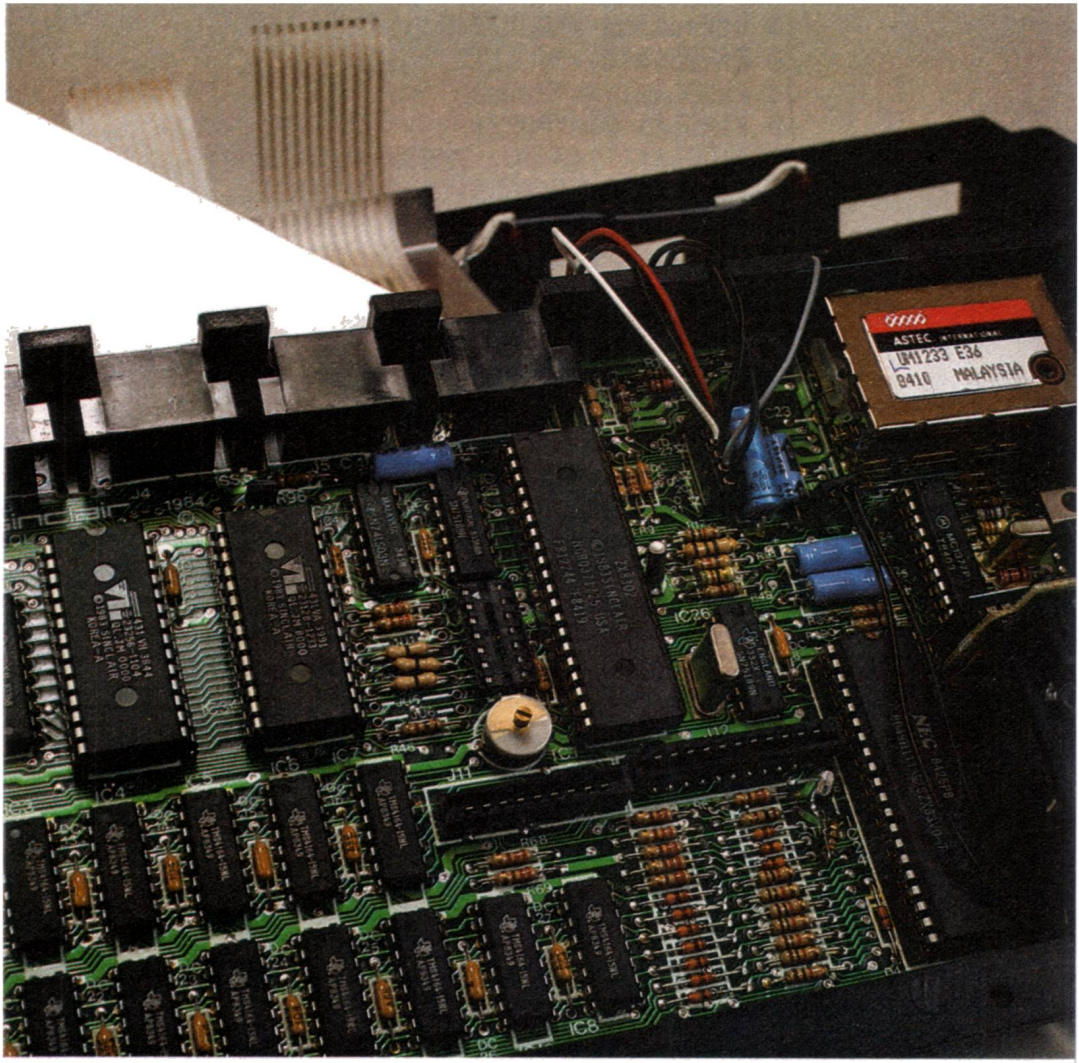
Usando le sottostringhe si puo' addirittura modificare parti di stringa, come viene mostrato nel programma che segue:

```
10 PRIMO$="SINCLAIR"
20 PRIMO$(3 TO 5)="BBC"
30 PRINT PRIMO$
```

Il risultato e' SIBBCAIR. Cioe' i caratteri 3,4,5 della stringa originaria ("NCL") sono stati sostituiti con un'altra stringa (BBC).

SOSTITUZIONE DI VETTORI

E' ovvio che la sovrapposizione di sottostringhe e' ammessa anche fra gli elementi di un vettore stringa. Si consideri,



Il particolare mostra gli attacchi per i collegamenti alla tastiera ed il modulatore TV funzionante in UHF.

ad esempio, il programma seguente:

```
10 DIMension PAROLA$(2,5)
20 PAROLA$(0)="JOLLY"
30 PAROLA$(1)="VERDE"
40 PAROLA$(2)="FORZA"
50 PAROLA$(1)(3 TO)=PAROLA$(2)(TO 2)
60 PAROLA$(0)=PAROLA$(2)(4)&PAROLA$(0)
70 PRINT PAROLA$(0)
80 PRINT PAROLA$(1)
```

Il risultato di questo strano programma e':

```
VFOE
ZJOLL
```

Che cosa e' successo alla "Y" alla fine di "JOLLY", stringa assegnata originariamente a PAROLA$(0)? In SuperBASIC, se non c'e' abbastanza posto per una stringa in un vettore, troncata (cioe' accorciata). Per il resto, PAROLE$(2) non viene modificata, mentre in PAROLA$(1) sono soggetti a variazione il terzo quarto e quinto carattere (3 TO), e precisamente il terzo e il quarto vengono sostituiti con i primi due (TO 2) di PAROLA$(2), mentre il quinto di prima rimane inalterato. Il risultato, percio', non puo' che essere la meravigliosa parola VEFOE (?).

Come si e' visto, se all'interno della stringa non c'e' posto per quella da inserire, la parte eccedente viene ignorata, viceversa se il posto e' eccessivo, viene utilizzato solo quello nesessario. Questo significa che non ci si deve preoccupare se le stringhe a destra e a sinistra del segno di uguaglianza (assegnazione) siano o meno della stessa lunghezza, pensa a tutto il QL.

SLICING DI VETTORI NUMERICI

Come annunciato precedendemente, in SuperBASIC si puo' applicare la tecnica dello slicing anche ai vettori numerici.

In questo caso, al contrario di prima, e' importante tenere sotto controllo le dimensioni degli elementi sottoposti a slicing. In effetti, anche se cio' non e' vero in generale, data al complessita' delle poche eccezioni, e' opportuno procedere come se fosse tassativo avere i due membri ai lati dell'uguale sempre delle stesse dimensioni, in modo che si possano sempre assegnare elementi di un vettore a quelli di un altro.

Con il programma che segue viene mostrato come si possano assegnare gli elementi di un vettore numerico ad un "sottovettore":

```
10 DIMension A(2,2)
20 DIMension B(4,4)
30 B(0,0)=7 : B(0,1)=99
40 B(0,2)=12 : B(1,0)=34
50 B(1,1)=11.4 : B(1,2)=6
60 B(2,0)=122 : B(2,1)=.9
70 B(2,2)=4346 : B(3,0)=971
80 B(3,1)=1 : B(3,2)=44
90 A=B(0 TO 2,0 TO 2)
```

Nell'ultima linea viene utilizzato lo slicing per assegnare tutti i valori del vettore A uguali a quelli del vettore B, quindi A(0,0)=7, cioe' a B(0,0), A(1,0)=34, cioe' B(1,0), e cosi' via.

LUNGHEZZA DI UNA STRINGA

La lunghezza di una stringa, ovvero il numero di caratteri di cui si compone, e' il valore che si ottiene dalla funzione LEN, come si puo' vedere dal seguente programma:

```
10 INPUT "COME TI CHIAMI ?";N$
20 PRINT:PRINT
30 PRINT "BENE,"!N$!"IL TUO NOME E' LUNGO"
40 PRINT LEN(N$)!"CARATTERI"
50 PRINT:PRINT
60 PRINT "'SALSICCIA' E' LUNGO"
70 PRINT LEN("SALSICCIA")!"CARATTERI"
```

CAPITOLO 8

PROGRAMMAZIONE STRUTTURATA

Uno dei maggiori pregi del SuperBASIC e' il contributo alla costruzione di programmi chiari e strutturati. Infatti, anche se in SuperBASIC sono ancora presenti i vecchi e poco eleganti meccanismi di salto (GOTO, GOSUB,ON...GOTO e ON...GOSUB), di fatto, sono stati inclusi per offrire un certo grado di compatibilita' con il BASIC degli altri calcolatori.

Come gia' e' stato detto, e' consigliabile per coloro che hanno dimestichezza con un altro BASIC (specialmente quello dello Spectrum), iniziare a programmare il QL piu' o meno come uno Spectrum, e, gradualmente, inserire gli elementi di SuperBASIC propri del QL.

Le parti di programma, che contengono istruzioni di controllo di sequenza, cicli e salti, sono quelle che ricaveranno maggiori vantaggi dal SuperBASIC.

STRUTTURE

Nell'ambito delle strutture il vocabolario del QL e' particolarmente ricco, a partire da

```
IF        THEN                [ELSE]       END IF
SELect    ON variabile=                    END SELect
FOR      [NEXT]               [EXIT]       END FOR
REPeat   [NEXT]                EXIT        END REPeat
```

per proseguire con i noti:

```
GOTO
GOSUB     RETURN
ON variabile GOTO/GOSUB numeri di linee
```

(anche se il loro uso non e' raccomandato), per finire con le procedure e le funzioni a piu' linee e con variabili locali:

```
DEFine PROCedure/FuNction nome [lista di parametri]
LOCAL variabili
RETURN valore
END DEFine
```

COMPATIBILITA'

La trattazione delle istruzioni di controllo della sequenza nei programmi inizia con le notissime istruzioni di GOTO, GOSUB/RETURN e ON...GOTO e ON...GOSUB/RETURN.

Ecco un semplice esempio del loro funzionamento:
6

```
10 PRINT "QUESTO E' IL QL ";
20 GOTO 10
```

come ci si aspettava, questo programma riempie lo schermo con la scritta fra virgolette, e continua a farlo finche' non lo si interrompe. Infatti il QL esegue la linea 10, poi si sposta alla linea 20, dove trova l'istruzione GOTO 10 (vai alla linea 10). Non vi sono condizioni poste a questo trasferimento di controllo dalla linea 20 alla linea 10, quindi il trasferimento avviene sempre e comunque. Questo e' un salto incondizionato.

GOSUB funziona in modo molto simile, ad eccezione del fatto che non e' mai solo, ma e' sempre accompagnato da RETURN, che dirige le operazioni di ritorno alla linea successiva a quella che conteneva GOSUB, come e' illustrato nel programma che segue:

```
10 PRINT "SONO ALLA LINEA 10"
20 GOSUB 50
30 PRINT "ORA SONO ALLA LINEA 30"
40 STOP
50 PRINT "SONO AL SOTTOPROGRAMMA DI LINEA 50"
```

```
60 PAUSE 50
70 PRINT "STO RITORNANDO"
80 PAUSE 50
90 RETURN
```

All'esecuzione del programma, sullo schermo del QL appaiono le seguenti scritte:

```
SONO ALLA LINEA 10
SONO AL  SOTTOPROGRAMMA DI LINEA 50
(una breve pausa)
STO RITORNANDO
(un'altra breve pausa)
ORA SONO ALLA LINEA 30
```

La parte di programma compresa tra la linea di controllo, il cui numero e' quello che segue GOSUB (linea 50), e quella che contiene RETURN (linea 90), e' detta sottoprogramma o subroutine.

Dunque analizzando l'esempio, il QL dice di trovarsi alla linea 10. Poi incontra il GOSUB 50 (linea 20), quindi passa alla linea 50, avvertendo con il messaggio "SONO AL SOTTOPROGRAMMA ...". Dopo una pausa, comunica che sta ritornando, ed infine dopo un'altra pausa, viene rispedito alla linea successiva a quella di partenza, cioe' la 30.

Il numero di linea "destinazione" di un GOTO o di un GOSUB puo' essere il risultato di operazioni aritmetiche (del tipo di X=A/B:GOTO X) o addirittura di espressioni (come in GOSUB 8XX). Ovviamente, si deve fare attenzione quando si usano espressioni come le precedenti, poiche' se si utilizza il comando di RENUM per rinumerare le linee del programma, il QL potrebbe essere mandato nel posto sbagliato da queste che, non essendo numeri, non vengono modificate.

Il QL possiede le varianti di GOTO e GOSUB, che non erano previste nei precedenti BASIC della Sinclair: le istruzioni ON...GOTO e ON...GOSUB.

Ed ecco subito un esempio di ON...GOSUB in azione:

```
 10 X=RND(1 TO 4)
 20 ON X GOSUB 40,60,80,100
 30 GOTO 10
 40 PRINT "QUELLO ERA UN UNO"
 50 RETURN
 60 PRINT "QUELLO ERA UN DUE"
 70 RETURN
 80 PRINT "QUELLO ERA UN TRE"
 90 RETURN
100 PRINT "QUELLO ERA UN QUATTRO"
110 RETURN
```

Il QL stampera' QUELLO ERA UN UNO se il numero casuale generato alla linea 10 (da RND) era un 1, QUELLO ERA UN DUE se era un 2, e cosi' via. Per individuare a quale delle destinazioni di GOSUB il calcolatore andra', basta numerare le destinazioni da sinistra verso destra. Cosi' la prima e' 40, cioe' il QL esegue GOSUB 40 se X e' uguale a 1. Il secondo e' 60, qundi per X=2 vale GOSUB 60, ... e cosi' via.

ON...GOTO funziona esattamente allo stesso modo, eccetto che, ovviamente, non esiste alcun RETURN per restituire il controllo della sequenza di esecuzione alla linea successiva.

RIDONDANZA

GOTO, GOSUB, ON...GOTO e ON...GOSUB non sono assolutamente necessari nel QL, poiche' in SuperBASIC vi sono strutture di controllo delle sequenze molto piu' raffinate, al punto che rendono le precedenti addirittura inutili e sconsigliate.

LA FRASE DI IF

La frase di IF e' il cuore delle capacita' del QL di prendere decisioni. In effetti, la principale differenza tra

una semplice calcolatrice e un computer e' proprio la capacita' di questi ultimi di decidere e di mettere in atto le decisioni.

Cosi' il QL, in comune con gli altri calcolatori, puo' valutare condizioni e utilizzare i risultati per orientarsi sui passi conseguenti da compiere.

Nel SuperBASIC vi sono tre versioni dell'istruzione IF. La piu' semplice ed anche la piu' nota e' IF/THEN.

IF (=se) il rubinetto perde THEN (=allora) si chiami un idraulico. IF sono stanco THEN vado a dormire. E cosi' via.

IF condizione vera THEN azione

Le due parole-chiave IF e THEN formano una sola istruzione. Ed ecco alcuni esempi:

```
IF NOME$ ="RINO" THEN PRINT "CIAO"

IF A+A>C THEN LET D=B

IF risposta=esatta THEN GOSUB 1000
```

Nella seconda forma, nella quale IF puo' essere usato sul QL, il THEN deve essere l'ultimo elemento della linea di IF. In questo caso le operazioni da eseguire dopo THEN possono essere su piu' linee, e si richiede al termine la parola chiave END IF.

Osservando il seguente frammento di programma, si vedra' come funziona il tutto:

```
...
100 IF R$="fine" THEN
110   PRINT "Grazie della partecipazione"
120   LET punteggio_finale=punti
130   LET conta_partite=conta_partite+1
140 END IF
...
```

Come si puo' vedere, tutte le istruzioni comprese fra IF/THEN (linea 100) e END IF (linea 140) vengono eseguite solo se la condizione alla linea 100 viene riscontrata vera.

La terza versione di IF/THEN consente due possibili percorsi In questo caso la struttura IF/THEN e' completata dall'aggiunta di ELSE (=altrimenti), che consente al QL di scegliere una seconda alternativa, se la prima non risulta possibile. Seguendo l'esempio tutto dovrebbe essere piu' chiaro:

```
...
100 IF punteggio>record THEN
110   PRINT "Nuovo record"
120   LET record=punteggio
130 ELSE
140   PRINT "Il record non e' stato battuto"
150 END IF
...
```

Il significato di questo frammento e' evidente: se il punteggio raggiunto e' maggiore del record, allora viene scritto "Nuovo record" e la variabile record viene aggiornata col nuovo valore, altrimenti (cioe' se la condizione non e' vera) compare la scritta "Il record non e' stato battuto".

THEN DIVENTA OPZIONALE

Diversamente dallo Spectrum, ma come in alcuni altri calcolatori, la parola-chiave THEN e' opzionale. Il QL, infatti, riesce a capire quando THEN e' stato omesso. Nella versione breve della frase di IF, THEN viene sostituito dal simbolo di due punti (:) come nell'esempio che segue:

```
IF risposta=esatta:PRINT "Giusto!"
```

Nella forma lunga della frase di IF, THEN puo' essere omesso del tutto:

```
IF nome$="Andrea"
  PRINT "E' un bel nome"
  LET punti=2*punti
END IF
```

NIDIFICAZONI

Le frasi di IF/END IF possono essere "nidificate", cioe' inserite le une nelle altre, tanto profondamente quanto si vuole, sempreche' ovviamente, non si usi tutta la memoria del QL in queste operazioni.
Il programma che segue e' un esempio di IF/END IF a due livelli. Come nei cicli nidificati di FOR/NEXT che si vedranno fra breve, ogni END IF si riferisce al precedente e piu' vicino IF:

```
100 IF risposta=esatta THEN
110   IF nome$=Andrea" THEN
120     PRINT "Ottimo, Andrea"
130   ELSE
140     PRINT "Hai ragione!"
150   END IF
160 END IF
```

RIPETIZIONI E CICLI

Nel SuperBASIC vi sono altri modi importanti di controllare la sequenza delle linee da eseguire: i cicli e le ripetizioni. Essi consentono, appunto, l'esecuzione ripetuta di parti di un programma.

Il primo di questi e' la frase di REPeat, che si struttura come segue:

```
REPeat identificatore
  istruzioni da ripetere
END REPeat identificatore
```

Esempio:

```
10 numero=RND(1 TO 100)
20 REPeat risposta
```

```
 30   INPUT "Qual'e' la risposta ?";risposta
 40   IF risposta=numero
 50     PRINT "Hai indovinato"
 60     EXIT risposta
 70   ELSE
 80     PRINT "Hai sbagliato, riprova"
 90   END IF
100 END REPeat risposta
```

Come si e' visto sopra, da un ciclo di REPeat si puo' uscire con l'istruzione di EXIT seguita dall'identificatore, percio' ogni frase di REPeat/END REPeat contiene almeno una istruzione di EXIT, poiche' altrimenti la ripetizione sarebbe continua e senza fine.

L'altra struttura di ripetizione e' quella di FOR/NEXT, ormai nota, solo che in SuperBASIC diventa ancora piu' potente.

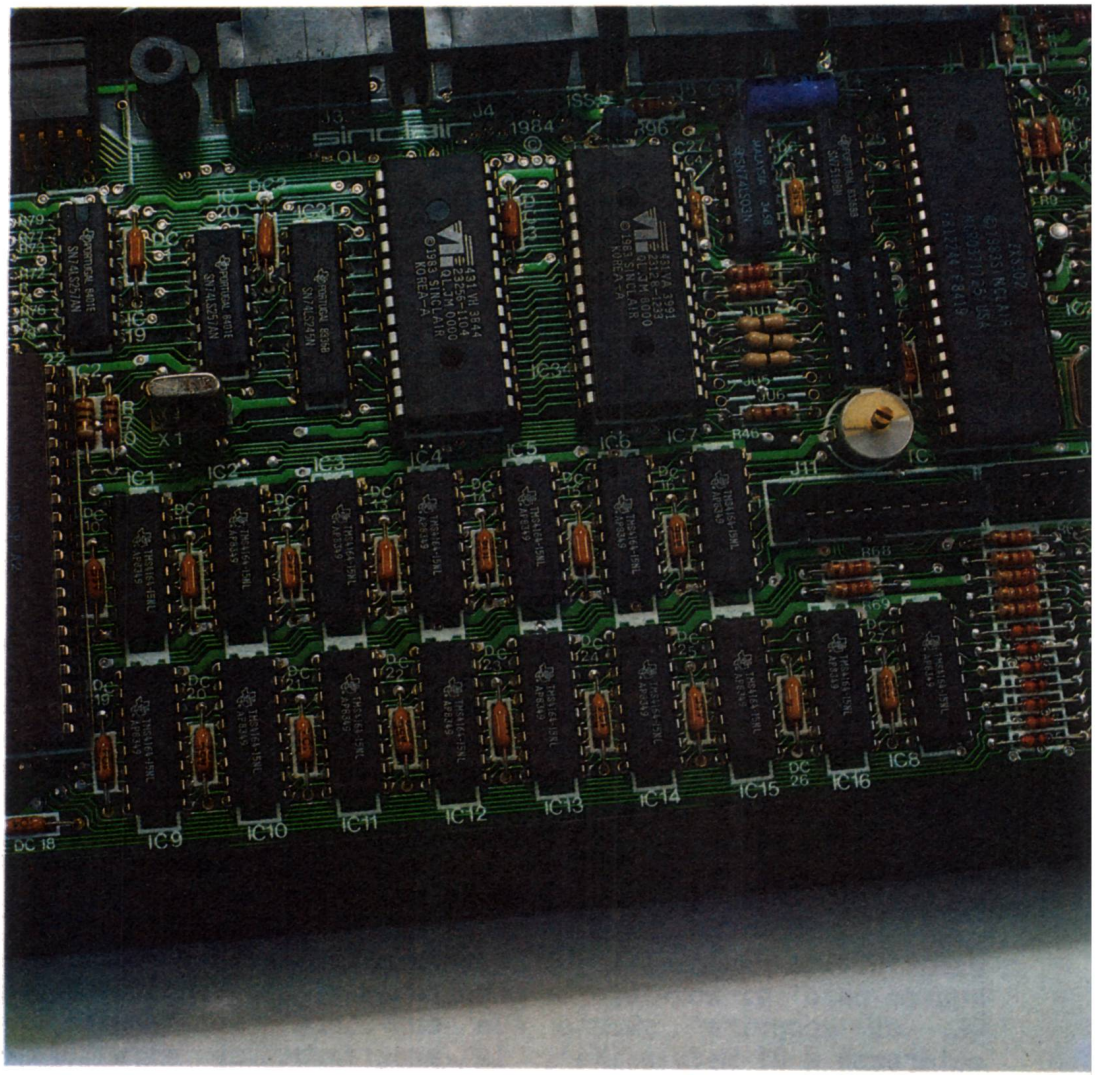
Anche questa istruzione puo' assumere due forme: una breve (ad una sola linea) e una lunga. In quella breve, NEXT (che in SuperBASIC e' spesso sostituito da END FOR) diventa opzionale , proprio come THEN nella forma breve di IF. La forma breve si costruisce come segue:

FOR var.=intervalli di ripetizione:istruz.da rip.:NEXT var.

mentre nella forma lunga:

```
FOR var_controllo=intervalli di ripetizione
  istruzioni da ripetere
  altre istruzioni da ripetere
  ...
END FOR var_controllo
```

In entrambi i casi, le istruzioni del ciclo vengono ripetute finche' la variabile di controllo assume un valore all'interno dell'intervallo di ripetizione, altrimenti si passa ad eseguire l'istruzione sucessiva alla fine del ciclo (END FOR o NEXT).



Sedici chip formano il banco di RAM da 128 K, 32 dei quali usati per il display. Sopra le RAM sono visibili le due EPROM da 16 K.

Si noti che il plurale "intervalli" non e' un errore di stampa, in SuperBASIC e' possibile elencare diversi intervalli di ripetizione come ad esempio:

```
FOR ciclo=1,2,3 TO 20,30,40 STEP 2
  ...
END FOR ciclo
```

Anche qui, come su molti altri calcolatori, si puo' specificare un passo (STEP) diverso da 1 per incrementare il valore della variabile di controllo ad ogni ciclo, e in piu' sul QL, si puo' omettere anche il TO , cioe' l'intervallo di ripetizione puo' essere descritto direttamente dai valori che deve assumere la variabile di controllo.

Altra particolarita' dei cicli di FOR in SuperBASIC e' quella di poter usare l'istruzione di EXIT, a patto di chiudere il ciclo con END FOR invece che con NEXT.

Gli aspetti piu' presigiosi di queste costruzioni sono: la possibilita' di definire forme estrmamente raffinate di controllo del tipo di DO/WHILE o REPEAT/UNTIL (presenti in alcuni BASIC avanzati); la capacita' di uscire prematuramente dal ciclo senza mandare in crisi il sistema (cosa che avviene su alcuni calcolatori, non appena si esce piu' di una volta dai cicli di FOR/NEXT senza completarli); ed infine la possibilita' di costruire senza recare offesa alcuna all'etica della programmazione strutturata.

Quest'ultima condizione puo' sembrare irrilevante per ora, quando pero' si acquistera' esperienza di programmazione sul QL, si scoprira' quanto sia superiore, in termini di chiarezza e di modularita', la programmazione strutturata rispetto a quella ordinaria. Anzi, dopo che ci si sara' conformati alle sue idee, diventera' naturale opporre resistenza a certi costrutti che, seppure funzionanti, sembreranno "tenuti assieme con lo spago".

Le strutture di FOR e REPeat possono essere completate da cio' che viene chiamato "epilogo" dalla Sinclair. Un epilogo e' una parte di programma posta, vicina alla fine di un

ciclo, che viene eseguita una sola volta e solo quando il ciclo e' terminato regolarmente.

Una possibile forma di epilogo e' la seguente:

```
FOR identificatore=intervalli di ripetizione
  istruzioni da ripetere
  IF condizione THEN EXIT identificatore
NEXT identificatore
  istruzioni di epilogo
END FOR
```

Dunque, l'epilogo viene eseguito solo se la condizione non e' mai verificata ed il ciclo principale termina regolarmente, se invece il ciclo ha termine con l'istruzione di EXIT, allora l'epilogo viene ignorato.

Come si e' visto, la condizione nella frase di IF e' determinante per l'esecuzione dell'epilogo.

SELEZIONE

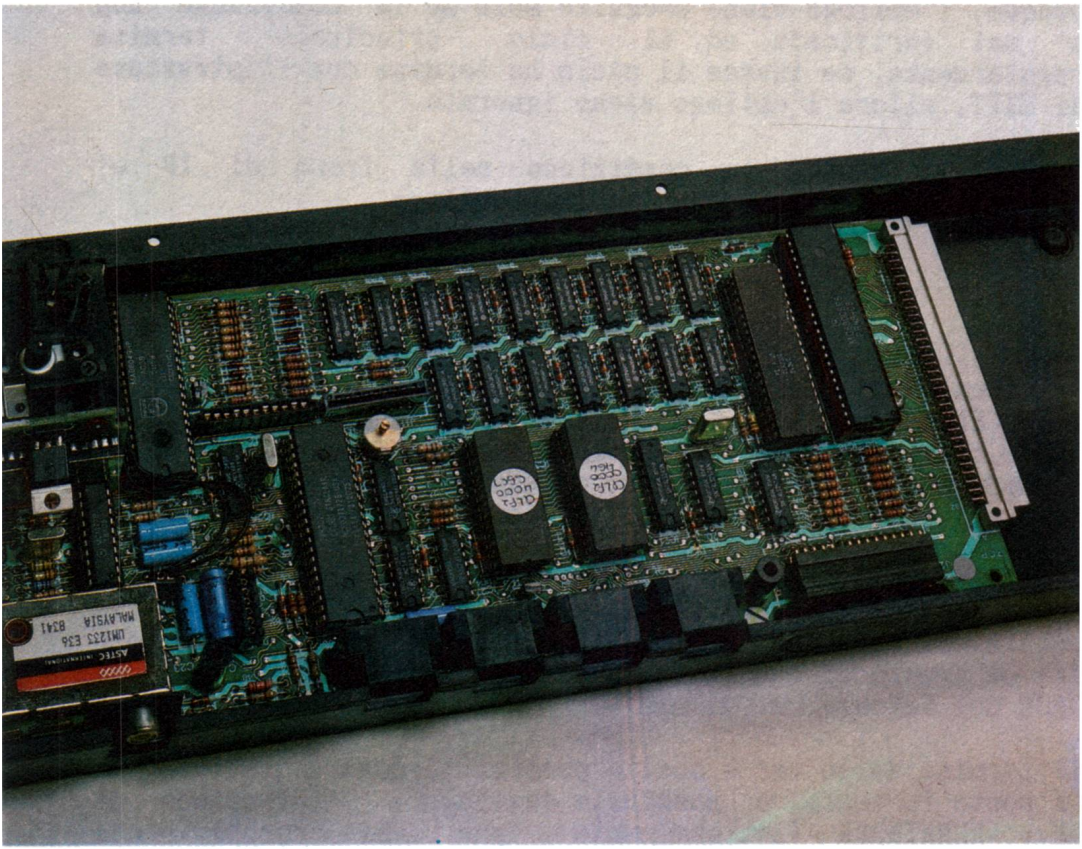
Nel vocabolario del QL sono previste le frasi di SELect...ON e END SELect, un altro meccanismo con il quale si determina cio' che deve essere fatto successivamente. Viene considerato il valore assunto da una variabile, ed in base a tale valore viene presa una decisione o un'altra.

A differenza di IF/THEN, dove si deve operare una scelta sulla base di una condizione, e di IF/THEN/ELSE, dove le condizioni sono due, SELect/END SELect consente di scegliere fra un numero di condizioni a piacere (come nella frase di CASE del Pascal).

Al termine delle varie scelte possibili, contraddistinte da ON posto in testa, e' possibile aggiungere l'istruzione di ON...=REMAINDER (letteralmente "per...=al resto"), che consente di effettuare l'ultima scelta per i restanti valori della variabile in esame.

L'esempio che segue chiarira' meglio la costruzione della frase di SELect/END SELect:

```
10 SELect ON livello_raggiunto
20   ON livello_raggiunto=0 TO 1999
30     PRINT "Non e' andata molto bene"
40     punti_bonus=-200
50     livello_raggiunto=0
60   ON livello_raggiunto=2000 TO 2999
70     PRINT "Sei stato promosso capitano"
80     punti_bonus=750
```



Posteriormente si aprono le porte di comunicazione con l'esterno. Si notano qui le RS232 e le CTRL port, nonchè, all'interno, il connettore per l'espansione di memoria.

```
 90      livello_raggiunto=0
100    ON livello_raggiunto=3000 TO 3999
110      PRINT "Ora sei il comandante"
120      punti_bonus=1500
130      livello_raggiunto=0
140    ON livello_raggiunto=REMAINDER
150      PRINT "Sei stato eletto Imperatore"
160      punti_bonus=3500
170 END SELect
```

COME SOSTITUIRE IF/THEN

SELect...ON puo' essere utilizzato, anche senza END SELect, per ottenere istruzioni piu' chiare di quelle con IF/THEN. Per dimostrarlo, ecco alcune frasi di IF/THEN

```
10 IF risultato>500 AND risultato<1000 THEN
     PRINT "Scarso"
20 IF risultato>999 AND risultato<2000 THEN
     PRINT "Sufficiente"
30 IF risultato>1999 AND risultato<3000 THEN
     PRINT "Buono"
```

modificandole con l'uso di SELect diventano

```
10 SELect ON risultato=501 TO 999:PRINT "Scarso"
20 SELect ON risultato=1000 TO 1999:PRINT "Sufficiente"
30 SELect ON risultato=2000 TO 2999:PRINT "Buono"
```

Anzi si puo' fare ancora meglio con:

```
10 SELect ON risultato
20   ON risultato=501 TO 999
30     PRINT "Scarso"
40   ON risultato=1000 TO 1999
50     PRINT "Sufficiente"
60   ON risultato=2000 TO 2999
70     PRINT "Buono"
80 END SELect
```

==========
C A P I T O L O 9
==========

FUNZIONI E PROCEDURE

Le funzioni e le procedure sono un altro punto di forza della programmazione del QL. Dopo che sono stete definite, per mezzo di DEFine FuNction e DEFine PROCedure, esse entrano in azione come comandi o in un programma quando vengono chiamate semplicemente con il loro nome.

Le funzioni sono formule che operano su variabili o costanti umeriche o stringa, dette argomenti, e forniscono un valore che dipende da quello assunto dagli argomenti.

La definizione di funzione richiede l'assegnazione di un nome seguito dall'elenco dei suoi argomenti, sempre racchiusi fra parentesi (con un nome fittizio per indicare il loro numero e ordine), le istruzioni di calcolo e la parola-chiave RETURN con l'identificatore del risultato. Si faccia attenzione, poiche' questo RETURN non e' lo stesso che si usa per indicare il termine di un sottoprogramma.

Anche se le definizioni di funzioni (e di procedure) possono essere poste in qualsiasi parte di un programma, e' consigliabile metterle all'inizio, in tal modo si rendera' piu' leggibile tutto l'insieme.

Ecco un esempio tipico di definizione di funzione :

```
10 DEFine FuNction MAX(x,y)
20   LOCAL risposta
30   IF x>y THEN
40     risposta=x
50   ELSE
60     risposta=y
70   END IF
80   RETURN risposta
90 END DEFine
```

Ora se si vuole utilizzare questa funzione in un programma, basta includere una linea come la 110 nel seguente frammento

```
100 a=...:b=...
110 PRINT MAX(a,b)
120 ...
```

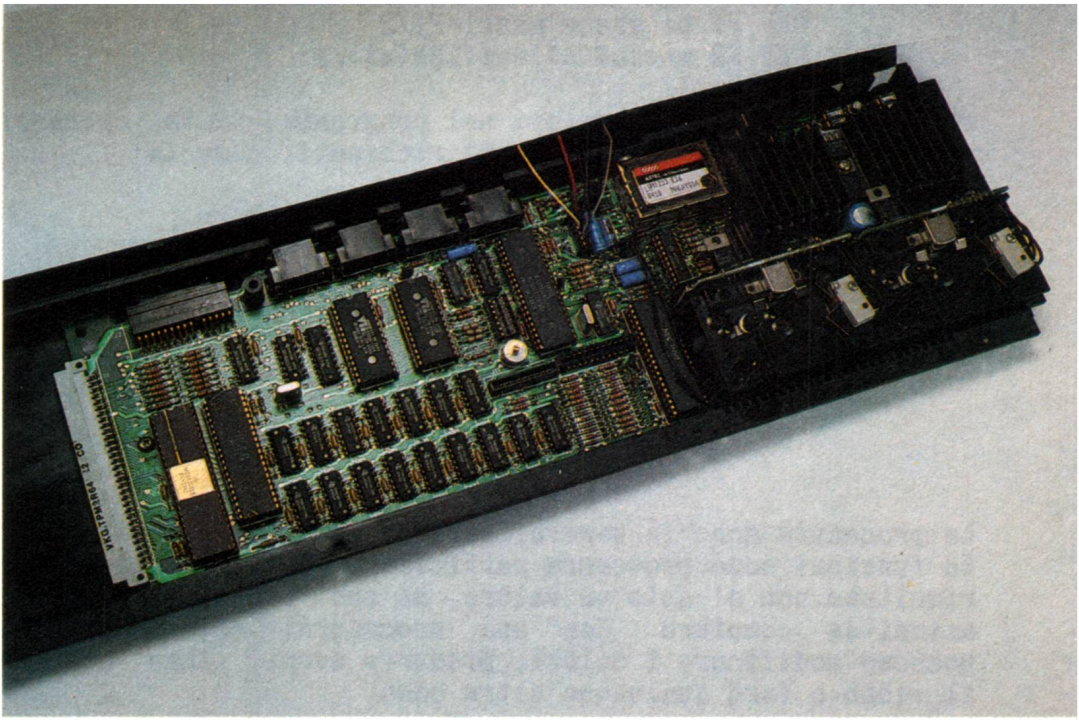
Come si puo' constatare, e' possibile aggiungere nuove funzioni matematiche (e non) al vocabolario del QL cosi' che, per il resto del programma, basta riferirsi al nome loro assegnato (MAX) per usufruirne , proprio come con quelle gia' disponibili del tipo SIN, COS, LEN e cosi' via.

Si noti l'uso della parola LOCAL nella linea 20 della definizione della funzione dell'esempio precedente. Essa assicura che il valore assegnato a "risposta" non verra' confuso con quello eventualmente presente in una variabile con lo stesso nome in un altro punto del programma. E' vivamente raccomandabile non assegnare mai valori a variabili non locali, all'interno di una funzione (o di una procedura), poiche' questo e' il modo piu' semplice di introdurre errori non facilmente individuabili nel programma.

Altro fatto importante per le funzioni (e le procedure) in SuperBASIC e' che esse possono essere ricorsive (!). Cioe' una funzione (o procedura) puo' richiamare se stessa. L'esempio proposto e' la classica funzione del fattoriale di un numero intero, cioe' il dato un numero intero n, la funzione calcola il prodotto dei primi n interi:

```
10 DEFine FuNction FAT(n)
20   LET n=INT(n)
30   IF n<2 THEN
40     RETURN 1
50   ELSE
60     RETURN nXFAT(n-1)
70   END IF
90 END DEFine
```

A linea 60, come si puo' vedere , la funzione FAT richiama se stessa con diverso valore dell'argomento (n-1).



Vista panoramica dell'interno: in primo piano la CPU 68008 prodotta dalla Motorola

In SuperBASIC e' anche possibile definire una funzione nella forma classica ad una sola linea, in tal caso non sono richieste ne' LOCAL, ne' RETURN e nemmeno END DEFine. Gli esempi che seguono sono definizioni di funzioni valide:

```
DEF FN arrot(x)=INT(x+0.5)
DEF FN media(x,y)=(x+y)/2
DEF FN a_caso(n)=RND(1 TO n)
DEF FN pi_greco_mezzi=PI/2
DEF FN mezzo$(a$)=a$(LEN(a$)/2)
```

Per utilizzare i loro valori nel programma, basta chiamarle con i loro nomi e gli argomenti richiesti, come in:

```
LET totale=arrot(totale)
IF media(a,b)>=6 THEN PRINT "Promosso"
FOR i=1 TO a_caso(100)
IF angolo=pi_greco_mezzi THEN PRINT "Retto"
PRINT mezzo$("prova")
```

PROCEDURE

Le procedure sono la generalizzazione delle funzioni, ovvero le funzioni sono procedure particolari. Con le procedure il risultato non e' solo un valore, ma puo' essere una serie di azioni da compiere. Con una procedura, ad esempio, si possono modificare i colori, produrre suoni, fare scorrere il video e fare qualunque altra cosa.

In generale, ogni azione effettuabile normalmente in SuperBASIC puo' essere fatta eseguire da una procedura. Si pensi che, fra le altre, la stessa istruzione di PRINT e' una procedura, anche se predefinita.

Ecco un semplice programma che dimostra come si definisce una procedura:

```
10 DEFine PROCedure quadrato(lato,x,y)
20   LINE x,y TO x+lato,
```

```
30   LINE TO x+lato,y+lato
40   LINE TO x,y+lato TO x,y
50 END DEFine
```

Con questa semplice procedura si richiede di tracciare un quadrato di lato lungo a piacere con l'angolo inferiore sinistro nel punto di coordinate (x,y) a scelta.

A differenza delle funzioni, quando si richiama una procedura, gli argomenti seguono il nome senza essere racchiusi fra parentesi. Quindi il comando

```
quadrato 50,20,20
```

provoca il disegno di un quadrato di lato 50 dal punto (20,20), oppure il programma:

```
10 FOR i=0 TO 20 STEP 5
20   quadrato 50-2*i,20+i,20+i
30 END FOR i
```

quando viene fatto eseguire, traccia 5 quadrati concentrici con lato decrescente di dieci unita' ognuno.

Anche le procedure, come le funzioni, possono essere prive di argomenti, ed in tal caso non sono richieste le parentesi nemmeno nella definizione dopo il loro nome, come nell'esempio che segue:

```
10 DEFine FuNction ora
20   LOCAL t$
30   LET t$=DATE$
40   PRINT t$(12 TO)
50 END DEFine
```

Con questa procedura, dopo aver attivato l'orologio del QL, si ottiene la stampa sullo schermo di ora:minuti:secondi non appena si batte sulla tastiera ora (e ENTER).

In conclusione, le procedure permettono di aggiungere nuove parole-chiave al vocabolario del QL, proprio come nel FORTH e nel PASCAL, altri linguaggi di programmazione di alto livello.

CAPITOLO 10

GRAFICA

Il QL e' provvisto di un vocabolario veramente ricco per quanto riguarda la grafica, dal controllo dello schermo, delle finestre video, dei bordi, al controllo del colore in alta e bassa risoluzione e altro. Fra le istruzioni grafiche si trovano:

AT	OVER
BLOCK	PAN
BORDER	PAPER
CLS	PRINT
CSIZE	RECOL
CURSOR	SCROLL
FLASH	STRIP
INK	UNDER

e WINDOW

Tutte queste parole-chiave (ad eccezione di PRINT) sono fatte seguire da parametri (numeri) che dicono al QL che cosa fare esattamente. In molti casi, tali numeri possono essere piu' di uno, come ad esempio uno o due dopo SCROLL oppure otto dopo RECOL. In generale, tutte queste istruzioni possono essere fatte seguire da #n (prima dei parametri), dove n e' il canale aperto per un particolare dispositivo di emissione anch'esso definibile a piacere. In assenza del parametro che indica il canale, l'istruzione si intende relativa a quello definito per difetto, cioe' il piu' delle volte lo schermo intero.

COLORI

Sul QL si hanno tre tipi di attributi di colore: il sottofondo (che viene indicato come colore principale), il contrasto ed il terzo tipo, prodotto utilizzando la tecnica

degli "stipples". I parametri di controllo dei colori possono, dunque, trattare uno, due o tutti e tre questi attributi.

CONTROLLO DEI COLORI

I parametri per questi attributi di colore vengono codificati in un byte: gli ultimi tre bit (0,1,2) contengono il colore principale, i secondi tre (bit 3,4,5) il colore di contrasto (ovvero lo XOR del colore principale) ed infine i due di testa (bit 6 e 7) il codice degli "stipples" scelti:

STIPPLE	CONTRASTO	PRINCIPALE
7 6	5 4 3	2 1 0

La definizione del colore puo' essere semplice, doppia o tripla, ottenibile cioe' con uno, due o tre parametri. Essa, dunque, e' semplice quando si usa un numero solo (da 0 a 255), in tal caso si fa riferimento alla rappresentazione binaria sopra esposta. In particolare, se si attivano solo gli ultimi tre bit (con un numero da 0 a 7) si richiede il colore solido senza stipples, come in PAPER 3.

La definizione e' doppia quando si specificano due parametri (da 0 a 7), in questo caso l'immagine risulta un miscuglio di colore principale e contrasto secondo una configurazione di stipples, assunta per difetto del tipo a scacchiera (vedere piu' avanti), come in PAPER 1,5.

Infine la definizione e' tripla quando si sceglie anche il tipo di stipples, specificando un terzo parametro (numero compreso fra 0 e 3).

STIPPLES

Gli "stipples" sono quadratini di quattro pixel, dei quali alcuni sono nel colore principale, altri in quello di

contrasto. Essi vengono riprodotti con scarso risultato sui televisori, per sfruttarne gli effetti in modo apprezzabile e' necessario disporre di un monitor. Comunque, come si potra' constatare sul proprio televisore, gli stipples possono essere utilizzati per ottenere colori diversi da quelli standard.

Vi sono quattro tipi di stipples numerati da 0 a 3, di essi il valore per difetto, quando cioe' non e' specificato diversamente, e' il 3 (quello a scacchiera). Negli schemi che seguono, gli zeri rappresentano il colore di contrasto, mentre gli uni quello principale:

STIPPLE	CONFIG.	DENOMINAZIONE
0	00 01	ad un solo pixel
1	00 11	a strisce orizzontali
2	01 01	a strisce verticali
3	01 10	a scacchiera

MODALITA'

Sul QL, come gia' e' stato anticipato, vi sono due gradi di risoluzione o modalita' grafiche. L'attivazione di una modalita' determina non solo la risoluzione, ma anche il numero di colori principali ottenibili. Date le elevate prestazioni grafiche, e' naturale che in alta risoluzione si possono ottenere immagini apprezzabili solo con un monitor, in caso contrario si avra' un certo "sfarfallio" dei pixel. E' importante rilevare inoltre che, quando si usa l'istruzione di MODE in un programa, tutte le finestre video (e i canali ad esse collegati) vengono "chiuse".

MODE 256 (o MODE 8): in questa modalita' lo schermo e' di 256 pixel in larghezza (numerati da sinistra verso destra da 0 a 511 poiche' ognuno e' contato due volte), per 256 in altezza (numerati verso il basso da 0 a 255) e i colori disponibili sono otto:

COD.		COLORE
0	-	nero
1	-	blu
2	-	rosso
3	-	magenta
4	-	verde
5	-	azzurro
6	-	giallo
7	-	bianco

Come con lo Spectrum, nel caso di un televisore o un monitor in bianco e nero, questi colori diventano otto tonalita' di grigio degradanti dal nero al bianco.

MODE 512 (oppure MODE 4): in questa modalita' lo schermo e' di 512 pixel in larghezza e 256 in altezza, e i colori disponibili diventano quattro:

COD.		COLORE
0 } 1 }	-	nero
2 } 3 }	-	rosso
4 } 5 }	-	verde
6 } 7 }	-	bianco

In realta' in alta risoluzione si riduce solo il numero dei colori principali, poiche' gli stipples consentono di creare un enorme spettro di colori (256 combinazioni). Ecco un programma (da usare solo con un monitor) che dimostra l'intero ventaglio dei colori disponibili sul QL:

```
 10 MODE 512
 20 FOR base=0 TO 7
 30   FOR contrasto=0 TO 7
 40     FOR stipple=0 TO 3
 50       PAPER base,contrasto,stipple
 60       PRINT "Base"!base!"contrasto"!contrasto!
 70       PRINT "stipple"!stipple
 80       PAUSE 50
 90     END FOR stipple
100   END FOR contrasto
110 END FOR base
```

Si ricorda che in questo programma, come in qualsiasi altro, se si vuole fermare l'esecuzione prima del termine, si deve premere CTRL e la barra spaziatrice.

PRINT

Una delle istruzioni piu' usate nella programmazione BASIC e' sicuramente quella di PRINT. Come negli altri BASIC, in SuperBASIC si usa PRINT per ottenere sullo schermo le informazioni emesse dal calcolatore, pero' in questo caso, si possono anche mandare le informazioni ad altri dispositivi, quali la stampante, altri calcolatori, ecc.

Lo schermo e' il dispositivo di emissione assunto per difetto, ed il canale ad esso associato e' #1, quindi con PRINT senza specificare il canale si ottengono le emissioni sullo schermo. Viceversa se si e' precisato in precedenza un dispositivo (ad esempio una stampante o una finestra video), e si e' aperto un canale di comunicazione tra il QL e questo dispositivo, per ottenere le informazioni sul dispositivo basta far seguire PRINT dal simbolo # e dal numero di canale.

La forma piu' semplice dell'istruzione di PRINT e' la seguente:

```
PRINT "Buongiorno, come va?"
```

Con questa istruzione si ottiene il messaggio (sopra indicato fra virgolette) direttamente stampato sullo schermo.

SEPARATORI

Le emissioni del QL includono l'uso di "separatori" che permettono di preparare le stampe secondo i formati desiderati, perfino anche nel caso in cui non si conosce la lunghezza degli elementi da stampare.

I separatori del SuperBASIC sono quattro:

! - il punto esclamativo e' usato come "spazio intelligente". Esso infatti inserisce uno spazio tra due elementi da stampare sullo schermo, oppure effettua un ritorno a capo alla riga successiva, se cio' che deve essere stampato non puo' essere contenuto per intero sulla riga corrente. Se poi lo spazio capita all'inizio di una riga, viene eliminato:

```
10 FOR elementi=1 TO numero
20   PRINT A$(elementi)!B$(elementi)!
30 END FOR elementi
```

, - la virgola e' il solito separatore che funziona da tabulatore spostando la posizione di stampa di otto in otto colonne (a partire dal'inizio della riga):

```
10 FOR conta=1 TO totale
20   PRINT elemento(conta),
30 END FOR conta
```

\ - la barra a rovescio effettua un ritorno a capo su una

nuova riga:

```
10 FOR numero=1 TO ultima
20   PRINT Cassa$(numero)!
30   IF INT(numero/5)=numero/5 THEN
40      PRINT \"---"\"XXX"\"---"
50   END IF
60 NEXT numero
```

; - il punto e virgola, infine, e' l'usuale separatore che permette di stampare il successivo elemento di seguito al precedente senza lasciare spazi:

```
10 FOR codice=32 TO 128
20   PRINT codice;" - ";CHR$(codice)
30 NEXT codice
```

Facendo eseguire i quattro brevi programmi precedenti (non senza aver prima definito le variabili contenute in essi), si potranno vedere gli effetti dei vari separatori.

AT

L'istruzione di AT sposta la posizione di stampa alla linea e colonna specificate dai due numeri che la seguono, separati fra loro da una virgola. Questi numeri variano da 0,0 per l'angolo in alto a sinistra, verso il basso e a destra in funzione delle dimensioni dei caratteri e della finestra video in uso. A differenza dello Spectrum, AT e' una vera e propria istruzione e quindi deve precedere l'istruzione di PRINT:

```
AT 12,8: PRINT "Qui sono alla linea 12 e colonna 8"
```

INK e PAPER

Sul QL, come sullo Spectrum, INK determina il colore delle stampe sullo schermo, mentre PAPER quello del sottofondo sul quale le scritte vengono stampate. Se si vuole modificare il

colore di sottofondo dell'intero schermo, dopo PAPER, si deve usare il comando CLS. Comunque la forma dell'istruzione in questo caso e' molto semplice, basta far seguire il numero del nuovo colore alla parola PAPER:

PAPER 3:CLS

questo comando, ad esempio, attiva come colore di sfondo il magenta in MODE 8 ed il rosso in MODE 4.

Il comando di INK invece attiva il colore di scrittura, ad esempio INK 1 produce scritte e grafici in blu in MODE 8 e in nero in MODE 4.

Una buona combinazione di colori, preferita dall'autore e dal traduttore, e'

PAPER 1:INK 7:CLS

che produce scritte in bianco su sfondo blu.

CLS

Come e' stato accennato piu' sopra, CLS viene usato per attivare il nuovo colore di PAPER su tutto lo schermo. La vera funzione di CLS pero' e' quella di ripulire lo schermo, ovvero la finestra video corrente, di tutto cio' che vi si trova.

CLS puo' essere fatto seguire da un numero da 0 a 4, per modificare la "globalita'" della pulizia come segue:

0 - Cancella tutto lo schermo, e' il valore per difetto, cioe' quando si omette il numero dopo CLS, viene sottointeso zero. Quindi CLS e CLS 0 sono del tutto equivalenti.

1 - cancella lo schermo dall'inizio fino alla linea contenente il cursore (esclusa).

2 - simile al precedente, cancella lo schermo dal fondo fino alla linea contenente il cursore (esclusa).

3 - cancella la linea contenente il cursore.

4 - cancella la parte rimanente della linea con il cursore a partire dal cursore stesso (incluso).

BORDER

Il comando di BORDER sul QL e' molto piu' flessibile del suo analogo presente sullo Spectrum. Il BORDER, ovviamente, e' l'area che circonda quella di PAPER. Con questo comando si possono scegliere sia il colore che la larghezza (in pixel) del bordo, quindi, ad esempio,si possono avere bordi di differenti colori e larghezze, uno dentro l'altro. Il programma riportato nel seguito produce un BORDER variopinto costituito da cornici di larghezza decrescente verso il centro dello schermo,l'area di PAPER risulta molto ridotta:

```
10 FOR largh=50 TO 2 STEP -2
20   BORDER largh,RND(0 TO 7)
30 END FOR largh
```

Per migliorare esteticamente le emissioni sullo schermo sono presenti altri comandi, dei quali viene data una breve descrizione.

FLASH

Le emissioni sullo schermo possono essere rese lampeggianti. Questo effetto e' ottenuto dal comando di FLASH 1, scambiando alternativamente fra loro i colori di PAPER e INK. Il programma che segue mostra come funziona la cosa:

```
10 PRINT "Non sto lampeggiando"
20 PAUSE 50
30 FLASH 1
40 PRINT "Ma io si!"
```

```
50 PAUSE 50
60 FLASH 0
70 PRINT "Io invece no"
```

INVERSE

Questo comando scambia semplicemente fra loro i colori di PAPER e INK:

```
10 INK 1:PAPER 2:CLS
20 PRINT "Questo e' rosso su blu"
30 INVERSE 1
40 PRINT "Adesso e' blu su rosso"
50 INVERSE 0
60 PRINT "Ed ora e' di nuovo come prima"
```

Ancora, come nel caso di FLASH, il valore 1 lo accende, mentre lo 0 lo spegne.

OVER e STRIP

Questi comandi lavorano insieme: OVER, seguito da -1, 0 oppure 1, determina il modo secondo il quale devono essere "sovrapposte" le emissioni; STRIP, seguito dai numeri di controllo del colore, produce un sottofondo, generalmente diverso da quello di PAPER, ed e' usato insieme con OVER 1, come un evidenziatore:

STRIP 1 seleziona uno sfondo blu, per i caratteri, quando si usa OVER 1.

OVER -1 effettua la stampa nel colore di INK sovrapponendola a quanto e' presente sullo schermo, senza cancellare.

OVER 0 disattiva la sovrapposizione, quindi le nuove scritte cancellano le precedenti e il colore di sottofondo e' quello di STRIP.

OVER 1 effettua la stampa nel colore corrente di INK sovrapponendola, senza cancellare, su un colore di fondo che e' quello di STRIP trasparente.

UNDER

Con questo comando si ordina al QL di sottolineare con il colore di INK corrente tutto quanto deve essere stampato:

```
10 INK 2:PAPER 1:CLS
20 PRINT "Scritta normale"
30 UNDER 1
40 PRINT "Scritta sottolineata"
50 UNDER 0
60 PRINT "Scritta non sottolineata"
```

Anche questo comando, come OVER e FLASH, deve essere seguito da 1 (attivato) oppure da 0 (disattivato).

PAN e SCROLL

Con questi comandi si possono ottenere effetti veramente affascinanti. A differenza dello ZX81, nel quale SCROLL effettua uno scorrimento dello schermo verso l'alto di una linea, sul QL lo scorrimento avviene nelle due direzioni e per pixel, consentendo cosi' un'ampia scelta. Inoltre, PAN, che non era presente negli altri calcolatori Sinclair, effettua lo scorrimento laterale, a destra o a sinistra, sempre di un numero di pixel a scelta.

PAN: se si fa seguire il comando solo da un numero positivo, si ottiene uno scorrimento verso sinistra del contenuto della finestra video corrente per il numero di pixel specificato. Ad esempio, PAN 24 fa scorrere verso sinistra quanto e' presente sullo schermo per 24 pixel, con l'aggiunta di spazio, a destra, nel colore di PAPER. Se invece il numero e' negativo, lo scorrimento avviene nell'altra direzione, cioe' PAN -65 sposta il contenuto

dello schermo verso destra di 65 pixel e completa lo schermo a sinistra con l'inserimento di spazi del colore di PAPER.

PAN, tuttavia, puo' essere fatto seguire da due numeri: il primo controlla il movimento dei pixel, come e' stato detto sopra, il secondo invece puo' essere 3 o 4. Se e' 3, come in PAN -25,3, viene effettuato lo scorrimento solo della linea contenente il cursore, in questo caso di 25 pixel verso destra. Se il secondo numero e' 4 , come in PAN -25,4, la linea del cursore viene fatta scorrere dalla posizione del cursore (incluso) fino al suo termine.

SCROLL: questo comando sposta il contenuto della finestra in su' o in giu'. In modo del tutto analogo a PAN si possono specificare la direzione e il numero di pixel per lo scorrimento. Anche qui, degli spazi occuperanno le aree della finestra rimaste vuote dopo lo scorrimento.

Dunque, se SCROLL e' fatto seguire da un solo numero positivo, il contenuto della finestra viene spostato verso l'alto, se negativo verso il basso.

Se, infine, si vuole fare scorrere solo una parte della finestra, basta aggiungere un secondo numero, separandolo dal primo con una virgola. Se si aggiunge 1 (come in SCROLL 19,1), viene fatta scorrere solo la parte di finestra fino alla linea precedente il cursore, con 2 si richiede lo scorrimento della parte inferiore della finestra a partire dalla linea successiva a quella contenente il cursore. In entrambi i casi la linea con il cursore non viene mai interessata dallo scorrimento.

CSIZE

Il SuperBASIC del QL possiede un'istruzione veramente eccezionale: CSIZE, abbreviazione di "Character SIZE, ovvero taglia dei caratteri. Essa consente di modificare in qualsiasi momento ed entro certi limiti le dimensioni dei caratteri da stampare sullo schermo.

La forma dell'istruzione e' la seguente:

CSIZE larghezza,altezza

All'accensione del QL, la taglia dei caratteri e' definita come 2,0 se si sta lavorando in MODE 256, e come 0,0 se in MODE 512.

Vi sono complessivamente quattro taglie in larghezza (da 0 a 3) e due in altezza (da 0 a 1). In verita' anche in larghezza le taglie sono solo due, le altre determinano una diversa spaziatura fra i caratteri. Il programma che segue mostra tutte le possibilita' offerte:

```
10 FOR h=0 TO 1
20   FOR l=0 TO 3
30     CSIZE l,h
40     PRINT "Questa e' la taglia";l;",";h
50   END FOR l
60 END FOR h
```

Nelle tabelle che seguono vengono precisate le dimensioni in pixel dei caratteri e le posizioni di stampa dello schermo in funzione dei codici di CSIZE:

CODICE LARGH.	PIXEL X CAR.	COLONNE MODE 256	COLONNE MODE 512
0	6	42	85
1	8	32	64
2	12	21	42
3	16	16	32

CODICE ALTEZZA	PIXEL X CAR.	NR. LINEE
0	10	25
1	20	12

CAPITOLO 11

SISTEMI DI COORDINATE

I progettisti del QL, prendendo spunto dal BBC Micro book, si sono assicurati che la posizione di ogni punto tracciato sul video non variasse in funzione della modalita' grafica in uso. Questo significa che si puo' usare MODE 512 con un monitor per ottenere della grafica ad alta risoluzione, e poi attivare MODE 256 per vedere gli stessi lavori su un televisore, senza dover ricalcolare tutte le posizioni relative dei punti rappresentati, lavoro quanto meno faticoso, se non estremamente complicato.

A questo scopo, nel QL sono presenti due sistemi di coordinate: il sistema grafico ed il sistema a pixel.

IL SISTEMA GRAFICO

Il "sistema di coordinate grafiche", detto piu' semplicemente sistema grafico pone l'origine (0,0) nell'angolo in basso a sinistra della finestra video corrente (se non specificato diversamente e determina (per difetto e automaticamente) l'unita' di misura lungo gli assi coordinati in modo che l'altezza massima sia di 100 unita'.

Gli assi, perpendicolari fra loro, sono orientati verso destra quello orizzontale e verso l'alto quello verticale come nella figura che segue:

```
        y ^
          |
          |
          |
          |
          |
          |
          |
          |
          |
(0,0)     |------------------------> x
```

Le dimensioni della finestra determinano la larghezza massima rappresentabile, mantenendo l'unita' di misura lungo l'asse delle x uguale a quella lungo l'asse delle y.

Fra le varie istruzioni (procedure predefinite) che utilizzano il sistema grafico, le principali sono:

- SCALE che determina il fattore di scala e l'origine delle coordinate grafiche nella finestra corrente.

- POINT che traccia un punto alle coordinate specificate.

- LINE che traccia un segmento tra due punti dati.

- ARC che traccia un arco di circonferenza fra due punti.

- CIRCLE che traccia una circonferenza o un ellisse.

- FILL che "riempie" le figure chiuse e convesse.

Come e' stato detto, quando si lavora con il sistema di coordinate grafiche, le unita' lungo l'asse verticale sono 100, salvo indicazioni contrarie. Con SCALE, appunto, si puo' modificare tale valore e contemporaneamente si sposta l'origine all'interno della finestra video corrente. Ad esempio, i valori per difetto sono quelli di SCALE 100,0,0.

Con POINT x,y si ottiene il punto di coordinate grafiche (x,y) rispetto all'origine grafica.

LINE, che serve per tracciare segmenti di linee rette, puo' assumere diverse forme, la piu' semplice delle quali e' la seguente:

```
LINE x1,y1 TO x2,y2
```

dove x1,y1 e x2,y2 sono le coordinate degli estremi del segmento.

La procedura ARC ha la seguente forma:

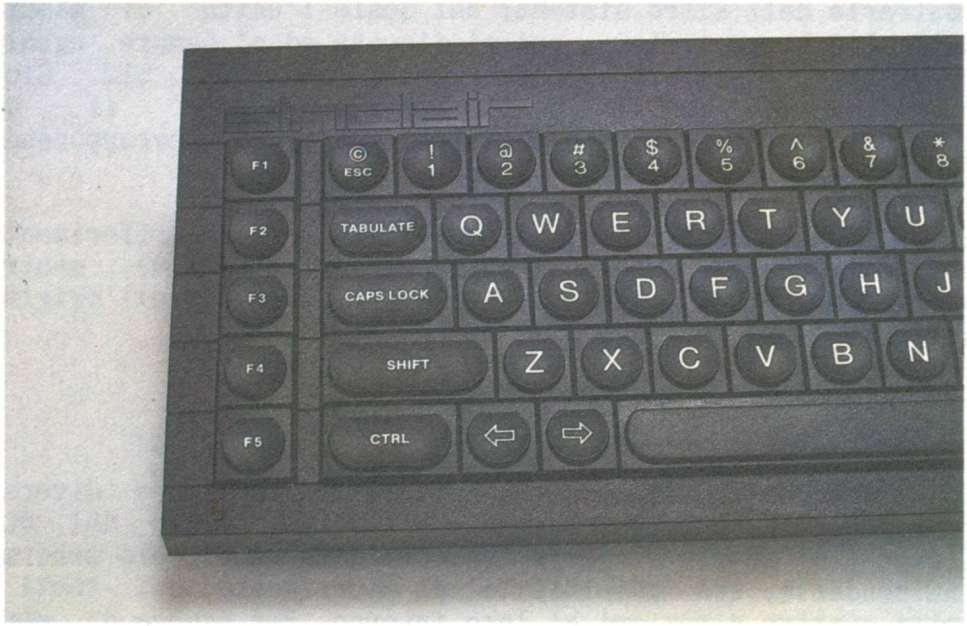
```
ARC x1,y1 TO x2,y2,radianti
```

dove x1,y1 e x2,y2 sono le coordinate degli estremi dell'arco e "radianti" e' l'ampiezza in radianti dell'arco. Le coordinate del primo estremo dell'arco possono essere omesse, in tal caso l'astremo sottinteso e' l'ultimo punto tracciato.

CIRCLE, se e' seguito da tre valori, come in CIRCLE x,y,r permette di tracciare una circonferenza con centro nel punto (x,y) e raggio r. Se invece e' seguito da cinque valori, come in CIRCLE x,y,r,e,a traccia un'ellisse di centro (x,y), asse maggiore lungo r, eccentricita' (rapporto tra asse maggiore e asse minore) uguale ad e, ruotata in senso antiorario di un angolo di a radianti.

FILL 1 attiva il riempimento (FILL 0 lo disattiva) di figure chiuse e convesse, cioe' senza rientranze, con il colore di INK corrente. L'istruzione di FILL deve precedere quelle di disegno della figura,come in:

```
FILL 1:CIRCLE 50,50,40
```



Particolare del lato sinistro della tastiera a membrana. Un lieve "click" denuncia la battitura del carattere evitando di consultare continuamente il video.

IL SISTEMA A PIXEL

Mentre il sistema a coordinate grafiche ha gli assi orientati rispettivamente verso destra e verso l'alto, quello a pixel li ha verso il basso e a destra come nella figura che segue:

```
(0,0) +---------------------+ (0,512)
      |                     y
      |
      |
      |
      |
      |
      |
(256,0) + x
```

Questo sistema di coordinate viene usato per determinare la posizione delle finestre e dei quadrati pieni e del cursore (ottenibili con i comandi WINDOW, BLOCK e CURSOR). Al contrario dell'altro sistema, nel quale l'unita' di misura delle lunghezze variava, qui e' fissata ed e' sempre uguale ad un pixel (=picture element) della modalita' 512. Cio' significa che se si lavora in MODE 256, il QL automaticamente adatta le posizioni dei punti, sovrapponendo lateralmente i pixel a due a due.

Si noti che alcuni comandi, come WINDOW, fanno riferimento all'origine dello schermo assunta per difetto (0,0), mentre altri, come BLOCK, sono definiti con riferimento all'origine della finestra in uso.

FINESTRE

Con l'uso delle finestre e' possibile definire diversi "minischermi" contemporaneamente attivi sul video del QL. Ciascuna di queste finestre funziona come un piccolo schermo indipendente, con possibilita' di scorrimento dei testi nelle quattro direzioni al loro interno. Per ognuna di esse e' anche possibile modificare PAPER e INK ed utilizzare CLS.

Per definire una finestra video si deve utilizzare una istruzione nella forma seguente:

OPEN #canale,SCR_larghezzaXaltezzaAgiu'Xdestra

dove SCR_ (abbreviazione di SCREEN schermo) e' il nome del dispositivo di emissione, larghezza e altezza sono in pixel, giu' e destra sono le coordinate dell'angolo superiore sinistro della finestra, e le X e A sono separatori.

Ad esempio

OPEN #5,SCR_120x100a32x16

definisce la finestra di canale 5, larga 120 e alta 100 pixel, a partire dal punto (32,16).

Il comando WINDOW serve a modificare la definizione delle finestre, e deve essere seguito da quattro parametri (espressioni numeriche) come segue:

WINDOW larghezza,altezza,x,y

dove larghezza e altezza sono valori in pixel, x e y sono le coordinate dell'angolo alto a sinistra della finestra.

Si noti che sia con OPEN che con WINDOW si deve usare il sistema di riferimento a pixel.

In SuperBASIC vi sono dei comandi che possono essere usati in relazione alle finestre e che accettano un paramentro finale supplementare che specifica l'azione del comando all'interno della finestra. I comandi, come gia' visto, sono CLS, PAN e SCROLL. Il parametro supplementare puo' assumere un valore da zero a quattro con i seguenti effetti:

0 - l'intera finestra
1 - sopra la linea del cursore (esclusa)
2 - sotto la linea del cursore (esclusa)
3 - solo la linea del cursore

4 - la parte a destra del cursore (incluso) della linea del cursore

Se si desidera gestire piu' di una finestra per volta, le finestre devono essere designate con il "numero di canale".

Quindi nell'istruzione WINDOW tale numero (sempre contraddistinto da #) deve precedere gli altri parametri:

WINDOW #numero,larghezza,altezza,x,y

Percio', quando ci si vuole riferire ad una particolare finestra, si deve aggiungere il suo numero di canale prima degli altri parametri. Cioe', ad esempio, per modificare il colore di INK nella finestra #3 si deve usare...

INK #3,5

...dove 3 e' il numero della finestra e 5 e' quello del colore di INK.

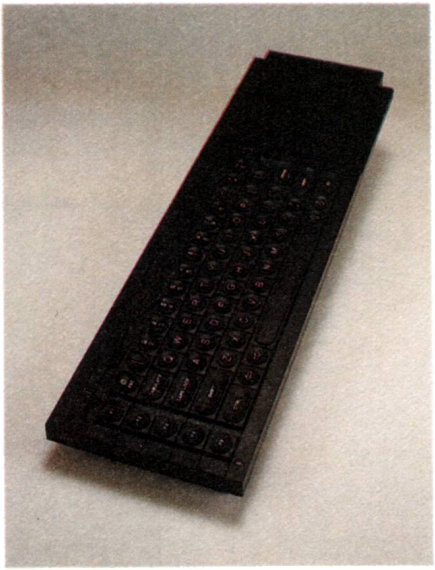
CURSOR

Con il comando di CURSOR si puo' posizionare il cursore dove si desidera all'interno di una finestra. Le dimensioni del cursore dipendono da quelle dei caratteri in uso.

CURSOR puo' essere seguito da due o quattro parametri, se usato con due parametri, essi sono le coordinate (nel sistema a pixel), rispetto all'origine della finestra, dell'angolo superiore sinistro del cursore. Se usato con due coppie di parametri, la prima coppia (in coordinate a pixel) determina l'origine delle coordinate grafiche, la seconda coppia indica le coordinate (nel sistema grafico) del cursore rispetto all'origine definita dalla prima coppia.

La forma piu' generale del comando e' dunque:

CURSOR #canale,giu',destra,x,y



Ecco gli accessori del QL:
l'alimentatore, il cavetto di comunicazione con lo Spectrum, il cavetto delle porte di controllo e tre piedini per inclinare il computer sul piano.

CAPITOLO 12

TRATTAMENTO DEI DATI

READ e DATA sono due parole-chiave che si accompagnano sempre nei programmi in quanto sono strettamente collegate fra loro, anche se, all'interno del programma, la loro distanza reciproca puo' essere grande.

READ/DATA

READ, che letteralmente significa "leggi", abilita la lettura ed il prelievo di dati posti al seguito della parola-chiave DATA. Ad esempio il breve programma che segue assegna dei colori agli elementi della schiera moda$:

```
10 DIMension moda$(5,6)
20 FOR colore=0 TO 5
30   READ moda$(colore)
40 END FOR colore
50 DATA "GIALLO","ROSA","NERO"
60 DATA "VIOLA","INDACO","ARANCIO"
```

Per verificare che i DATA (elementi nelle linee 50 e 60) sono stati assegnati alla schiera moda$ dall'istruzione READ, si possono aggiungere le tre linee seguenti e poi far rieseguire il programma:

```
70 FOR colore=0 TO 5
80   PRINT moda$(colore)
90 END FOR colore
```

Come si puo' vedere, il QL, quando incontra le linee 50 e 60, le ignora. Le linee di DATA, infatti, sono solamente lette quando e' necessario, altrimenti e' come se non esistessero.

La posizione nel programma delle frasi di DATA e' del tutto indifferente, come viene mostrato dall'esecuzione sul QL del programma che segue:

```
 10 DATA "Ruston"
 20 DIMension autore$(6,7)
 30 FOR libro=0 TO 6
 40   DATA "Hartnell","Nelson"
 50   READ autore$(libro)
 60   DATA "Gifford","Hole"
 70 END FOR libro
 80 DATA "Shaw","Vincent"
 90 FOR numero=6 TO 0 STEP -1
100   PRINT autore$(numero)
110 END FOR numero
```

In questo caso i dati vengono comunque trovati anche se sono sparpagliati qua e la' (alcuni addirittura dentro il ciclo di FOR/END FOR), infatt si ottiene:

```
Vincent      Hole         Nelson
Shaw         Gifford      Hartnell      Ruston
```

RESTORE

La terza parola-chiave che viene usata congiuntamente a READ e DATA e' RESTORE. Con questo comando si ordina al calcolatore di ripristinare la posizione del puntatore dei dati, cioe' di riportare l'attenzione all'inizio del blocco di DATA, come mostra questo programma:

```
 10 DIMension insomma$(7,7)
 20 FOR perdinci=0 TO 7
 30   IF perdinci=4 THEN RESTORE
 40   READ insomma$(perdinci)
 50   PRINT insomma$(perdinci)!
 60 END FOR perdinci
 70 DATA "Questo","e'","un","esempio"
 80 DATA "di","come","funziona","RESTORE"
```

Ad un affrettato esame del programma, si potrebbe supporre di ottenere il seguente risultato:

Questo e' un esempio di come funziona RESTORE

Invece sullo schermo apparira':

Questo e' un esempio di Questo e' un

Infatti quando la variabile "perdinci" e diventata uguale a quattro, RESTORE riporta il puntatore dei dati indietro allo inizio delle DATA, e quindi l'istruzione di READ riprende la lettura daccapo.

RESTORE puo' anche rimandare ad una linea particolare, come si puo' vedere dal prossimo programma:

```
 10 FOR gusto=1 TO 8
 20   IF RND(1 TO 8)=4 THEN RESTORE 100
 30   READ pizza$
 40   PRINT pizza$
 50 END FOR gusto
 60 DATA "Margherita","Al prosciutto"
 70 DATA "Quattro stagioni","Napoletana"
 80 DATA "Calzone","Pugliese"
 90 DATA "Ai frutti di mare"
100 DATA "Capricciosa","Boscaiola"
110 DATA "Al gorgonzola","Al trancio"
120 DATA "Alle acciughe","Alle cipolle"
130 DATA "Alla BismarK","Fantasia"
```

In questo programma, se il numero casuale generato alla linea 20 e' uguale a 4, allora il puntatore dei dati viene spostato alla linea 100, e da quel punto parte la lettura dei dati. Questo significa che, nell'ipotesi che il quattro esca dopo cinque letture, il risultato sarebbe:

```
Margherita                Calzone
Al prosciutto             Capricciosa
Quattro stagioni          Boscaiola
Napoletana                Al gorgonzola
```

Un uso molto diffuso di istruzioni DATA e' quello della introduzione di valori specifici negli elementi dei vettori e delle schiere nella fase di inizializzazione dei programmi.

CAPITOLO 13

ESTENSIONE DEL VOCABOLARIO

In questo capitolo si e' cercato di illustrare alcuni dei comandi del SuperBASIC che per importanza o utilita' arricchiscono il vocabolario del QL.

NUMERI CASUALI

I numeri casuali vengono spessissimo usati nei programmi di giochi e di simulazione. Il QL dispone di un sistema davvero unico con il quale specificare il tipo di numeri casuali desiderati.

Con il comando RANDOMIZE si inizializza il generatore di nemeri casuali e lo si fa partire da un numero dato o da un valore preso dal QL al suo interno. Nel primo caso il numero di "innesco" racchiuso fra parentesi segue RANDOMIZE, nel secondo caso non serve altro, e' sufficiente il comando senza parametri.

Per ottenere i numeri casuali si deve usare la parola RND, abbreviazione di RaNDom, in una delle forme che seguono, di cui la prima e piu' semplice e' quella senza parametri:

```
10 PRINT RND!
20 GOTO 10
```

All'esecuzione delle due linee di cui sopra si ottiene una serie apparentemente infinita di numeri tutti diversi fra loro, compresi fra zero e uno (estremi esclusi). L'avverbio apparentemente si riferisce al fatto che dopo molto tempo la serie si ripete, anche se e' improbabile che si raggiunga tale situazione.

Una seconda forma di RND puo' essere usata quando si desiderano dei numeri casuali interi compresi fra zero e un limite superiore prestabilito (incluso). In tal caso l'esempio diventa:

```
10 PRINT RND(limite_superiore)!
20 GOTO 10
```

E, se si suppone che il limite_superiore sia 7, si potrebbe ottenere la seguente successione di nemeri:

```
6 0 5 2 6 1 3 2 5 1 0 3
4 3 1 5 0 2 6 1 5 1 7 4  ...
```

Puo' sicuramente capitare di aver bisogno di numeri casuali compresi in un intervallo diverso da quello fra zero e uno. In questo caso e' stato previsto l'uso di TO parola che la Sinclair ha introdotto ad arte nel suo BASIC:

```
RND (limite_inferiore TO limite_superiore)
```

Ad esempio, si potrebbero richiedere dei numeri casuali compresi fra 95 e 105, allora si avrebbe:

```
10 PRINT RND(95 TO 105)!
20 GOTO 10
```

che potrebbe dare i seguenti risultati:

```
104 96 105 101 95 97 101
98 103 105 99 102 100 96 ...
```

ALTRI COMANDI

Fra i comandi di utilita' generale in SuperBASIC vi sono AUTO, DLINE e RENUM.

AUTO e' utilizzato per generare automaticamente i numeri di linea. Puo' essere seguito da due, uno o nessun parametro,

In dotazione, oltre al manuale d'istruzione, anche due confezioni da quattro microdrives.

la sua forma generale e' la seguente:

AUTO inizio,passo

I suoi valori per difetto sono 100 e 10, cioe' in assenza di parametri la numerazione delle linee inizia da 100 e prosegue di 10 in 10. Per bloccare la autonumerazione si deve premere CTRL insieme con la barra spaziatrice.

DLINE serve ad eliminare una o piu' linee di un programma ed e' seguito dai numeri o dagli intervalli di numeri di linea da eliminare. la sua forma e':

DLINE linea/inizio TO fine,...

dove la barra (/) indica possibilita' alternativa e i puntini possibilita' di ripetizione.

Quando manca il numero di linea iniziale si sottintende l'inizio del programma, quando manca quello finale si sottintende la fine del programma. Ad esempio se si vogliono eliminare la linea 30, quelle dalla 120 alla 200 (incluse) e dalla 500 alla fine si scrivera':

DLINE 30,120 TO 200,500 TO

RENUM effettua la rinumerazione parziale o totale delle linee del programma, la sua forma e' la seguente:

RENUM inizio TO fine;nuovo inizio,passo

Se RENUM non e' seguito da alcun parametro, tutto il programma viene rinumerato con nuovo inizio uguale a 100 e passo 10. Si faccia attenzione poiche' i numeri di linea che seguono RESTORE vengono ingorati da RENUM, ed inoltre, se vi sono GOTO e GOSUB seguiti da espressioni che iniziano con un numero, RENUM lo modifica come se fosse un numero di linea.

CLEAR e NEW

CLEAR elimina dalla memoria tutte le variabili con i loro valori, restituendo la disponibilita' dello spazio da esse

occupato. Quando si esegue RUN, automaticamente viene effettuato un CLEAR.

NEW combina l'azione di CLEAR con l'eliminazione dei programmi presenti nel QL, quindi deve essere usato con estrema cautela.

PAUSE

Il comando di PAUSE serve a fermare l'esecuzione di un programma per un tempo prestabilito, in attesa della scadenza o della pressione di un tasto. Facendo seguire PAUSE da un numero si specifica il tempo massimo di attesa in cinquantesimi di secondo. Ad esempio con PAUSE 100 si richiede una pausa di 100/50 cioe' 2 secondi. Se non viene indicato il tempo, la pausa e' a tempo indeterminato, cioe' fino alla pressione di un tasto.

REM

Quando, nel corso della stesura di un programma, si vogliono inserire dei commenti per promemoria o per documentazione senza che il calcolatore ne tenga conto, si usano frasi che iniziano con REMark, come ad esempio:

```
10 REMark Questo programma prova RND
```

Una frase di REMark puo' seguire una normale istruzione come in:

```
100 PRINT "Hai vinto!":REMark Fine del gioco
```

Naturalmente non si deve iniziare un linea a piu' istruzioni con una frase di REMark poiche' verrebbe ignorata dal QL, come nella linea che segue, nella quale non verra' mai eseguita la seconda istruzione:

```
450 REMark Fine del gioco:PRINT "Hai vinto!"
```

LIST

Il comando di LIST consente di ottenere sullo schermo (o su qualche altro dispositivo/canale) l'elenco delle istruzioni del programma presente in memoria.

A parte il numero di canale (preceduto da #) che puo' mancare, il comando puo' essere usato in cinque modi diversi:

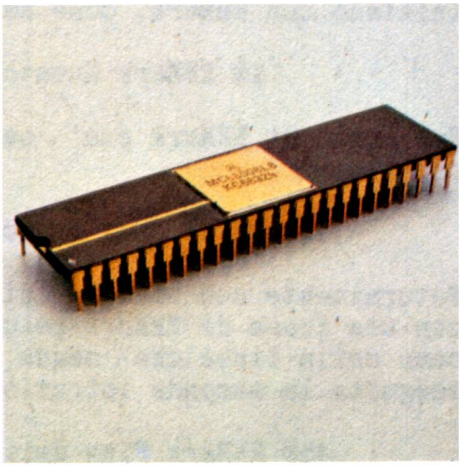
LIST 100 TO 200 mostra il listato a partire dalla linea 100 fino alla 200 inclusa.

LIST 100 TO mostra la lista dalla linea 100 fino al termine del programma

LIST TO 100 mostra le linee dall'inizio del programma alla 100 inclusa

LIST 100 lista solo la linea 100

LIST fornisce l'intero listato del programma



Il cervello del QL è il microprocessore a 32 bit della Motorola 68008 dotato di ben 48 terminali.

PEEK e POKE

PEEK e POKE consentono di guardare in una locazione di memoria (PEEK) o di modificarne il contenuto (POKE). La loro forma e' la seguente:

```
PEEK (indirizzo)
POKE indirizzo,numero
```

Queste istruzioni implicitamente specificano che l'accesso alla memoria deve avvenire per byte (8 bit), quindi viene preso in considerazione solo il byte di indirizzo indicato. Le parentesi nel caso di PEEK sono obbligatorie. Se invece si scrivono nella forma:

```
PEEK W (indirizzo)
POKE W indirizzo,numero
```

si richiede che l'accesso sia a parole (word=16 bit), cioe' vengono considerati i due byte consecutivi di indirizzo e indirizzo+1. Esiste infine una terza forma di queste importanti istruzioni:

L'espansione di memoria viene effettuata tramite l'apposita cartuccia che eleva la capacità da 128 K a 640 K.

```
PEEK_L (indirizzo)
POKE_L indirizzo,numero
```

in quest'ultima forma l'accesso e' a "parole lunghe" (32 bit), cioe' vengono considerati quattro byte consecutivi.

STOP

Ovviamente il comando di STOP e' usato per arrestare definitivamente l'esecuzione di un programma:

```
110 ...
120 IF risposta$="Costantinopoli" THEN STOP
130 PRINT "Sbagliato!"
140 ...
```

INPUT

INPUT serve a richiedere dati in immissione nel corso della esecuzione di un programma. Dopo INPUT e prima delle variabili destinate a contenere dati, sono ammessi dei messaggi come in:

```
INPUT "Come ti chiami?"!nome$
INPUT "Quanti anni hai?"!anni
```

Sono anche ammesse piu' richieste di dati in una sola frase di INPUT, in ogni caso si possono usare i separatori per migliorare i formati di stampa come ad esempio in:

```
INPUT "Quanti?"!numero,"peso?"!kg
INPUT "Cognome?"!c$,"Nome?"!n$
```

INKEY$

Nel corso di un programma puo' essere necessario fare leggere direttamente al calcolatore la tastiera (o qulsiasi

altro canale specificato) e prelevare il contenuto del primo tasto premuto (o del primo elemento inontrato). E' questo il caso di INKEY$:

```
10 IF INKEY$<>""THEN GOTO 10
20 IF INKEY$="" THEN GOTO 20
30 IF INKEY$="a"THEN PRINT "OK"
```

Questo breve programma funziona come segue: prima fa leggere la tastiera ripetutamente finche' non e' stata liberata da precedenti pressioni di tasti (quindi ripete l'esecuzione della linea 10 finche' INKEY$ e' diverso dalla stringa vuota);poi,quando la tastiera e' trovata libera, il QL passa alla linea 20 e vi rimane finche' non viene premuto un tasto; infine il programma arriva alla linea 30 e,se il tasto premuto e' quello della lettera a minusola (cioe' INKEY$="a"), allora viene scritto "OK".

Questa istruzione, in SuperBASIC, e' dotata di un ulteriore opzione: quella di poter scegliere il tempo di attesa massimo del tasto premuto. Per ottenere questo si deve far seguire INKEY$ da un numero fra parentesi con i seguenti significati:

INKEY$(n)	attesa di n cinquantesimi di secondo
INKEY$(-1)	attesa a tempo indeterminato (fino alla pressione di un tasto)
INKEY$(0)	nessuna attesa (valore per difetto)

TARTARUGA GRAFICA

Il SuperBASIC dispone di un insieme di comandi grafici, solitamente presenti in altri liguaggi quali il LOGO, che controllano quella che comunemente viene chiamata tartaruga grafica.

Si tratta di un punto sullo schermo (o sulla finestra video specificata) controllabile come se fosse una penna. I comandi disponibili sono riassunti nello schema che segue:

Comando	Azione	parametro
PENUP	smette di disegnare	nessuno
PENDOWN	inizia a disegnare	nessuno
MOVE	percorre un cammino	distanza
TURN	ruota di un angolo	angolo in gradi
TURNTO	prende la rotta	angolo in gradi

Inizialmente la tartaruga e' posizionata sul lato destro dello schermo, orientata a zero gradi e non lascia traccia. Il parametro distanza puo' essere positivo o negativo: se positivo la tartaruga avanza, se negativo indietreggia. Anche gli angoli possono essere negativi, in tal caso le rotazioni avvengono in senso orario.

CAPITOLO 14

QDOS e CANALI

QDOS e' il nome che e' stato dato al sistema operativo del QL. I suoi compiti sono tanti ed importanti. In particolare deve sorvegliare i sei principali sottosistemi interni del calcolatore:

- stabilisce quale parte del calcolatore deve svolgere questo o lavoro e quando;
- controlla le emissioni sullo schermo;
- dirige le operazioni dei Microdrive;
- effettua la supervisione del QLAN (la rete locale) e delle interfacce RS-232-C;
- legge la tastiera;
- amministra la distribuzione della memoria.

LE MAPPE DELLA MEMORIA

Nel QL si possono distinguere due mappe della memoria. La prima e' basata sull'hardware, cioe' sul supporto fisico, mentre la seconda e' l'organizzazione della memoria RAM imposta dal QDOS. Si noti che quest'ultima e' soggetta ai mutamenti derivati dagli sviluppi di nuove versioni del QDOS, che la Sinclair produce e distribuisce.

MAPPA FISICA DELLA MEMORIA

FFFFF	Vettori interruzioni	solo 7 determinano lo stato del processore
FFFF3		
	RISERVATA	solo per macchine con espasione RAM
40000		
3FFFF	RAM 96K	
28000		
27FFF	RAM 32K	schermo/video
20000		
1FFFF	RISERVATA	dispositivi ed espansioni I/O
10000		
0FFFF	ROM 32K	ROM connessa
08000		
07FFF	ROM 32K	ROM di sistema
00000		

MAPPA DELLA MEMORIA RAM

SV_RAMT-1	procedure residenti	Questa area si espande verso il basso
SV_RESPR	programmi transienti	
SV_TRNSP	interprete comandi SuperBASIC + programmi e dati SuperBASIC	
SV_BASIC	sottosistema di registrazione dei blocchi slave	Quest'area assorbe la memoria disponibile
SV_FREE	canali e altre informazioni	Quest'area si espande verso l'alto
SV_HEAP	controllo risorse+ tavole e variabili di sistema	28000 esadecimale
	memoria video	20000 esadecimale

CANALI

Quando si lavora sul QL, come si e' gia' avuto modo di vedere in precedenza, capita spesso di usare i canali. L'argomento e' molto importante nel QL e percio' e' opportuno un chiarimento su cio' che si intende per canale in SuperBASIC.

I canali molto semplicemente sono dei condotti collegati con il QL ad una estremita' e con altri oggetti (i dispositivi) all'altra. Lungo i canali le informazioni scorrono avanti e indietro, entrando e uscendo dal QL e dai dispositivi collegati.

Per poterli usare questi canali devono essere aperti con il comando OPEN, e al termine dell'uso vanno chiusi con CLOSE. Alcuni di essi vengono automaticamente aperti all'accensione del QL, e poiche' non si devono specificare, sono detti "canali per difetto". Per la stessa ragione i dispositivi a cui questi canali conducono si chiamano "dispositivi per difetto".

Ad esempio, lo schermo e' un dispositivo per difetto. Infatti se non si specifica che le emissioni da effettuare con PRINT devono essere inviate da qualche altra parte, attraverso un diverso canale, esse compariranno sicuramente sullo schermo. Un canale per difetto non puo' mai essere chiuso.

I canali vengono identificati assegnando loro un numero (preceduto da #). Una volta assegnato un numero (tra 0 e 15) ad un canale, quel numero e' tutto cio' che serve al QL per sapere quale dispositivo si desidera usare.

DISPOSITIVI DI I/O

La descrizione dei dispositivi di ingresso e uscita (I/O) viene effettuata con l'apertura di un canale, facendo seguire l'istruzione di OPEN #canale dal nome del

dispositivo e dai parametri e separatori necessari alla descrizione del dispositivo. Nel seguito viene riportato l'elenco dei dispositivi di I/O con i relativi parametri (i nomi dei dispositivi e i separatori alfabetici sono indicati in maiuscolo per distinguerli dai parametri, mentre le parenesi quadre indicano parametri opzionali senza separatori):

SCR_larghXaltAxXy	finestra video
CON_lsrghXaltAxXy	console
SER[num][par][hands][eof]	interfacce RS 232 C
MDVnum_nome	microdrive
NETtipo_numdest	rete locale

CONTROLLO OPERAZIONI DI I/O

Segue un sommario delle parole-chiave che controllano le operazioni di I/O:

```
BAUD numero
CLOSE #canale
COPY disp TO disp
COPY N disp TO disp
DELETE disp nome
DIR disp nome
NET numstazione
OPEN #canale,disp
OPEN IN #canale,disp
OPEN NEW #canale,disp
```

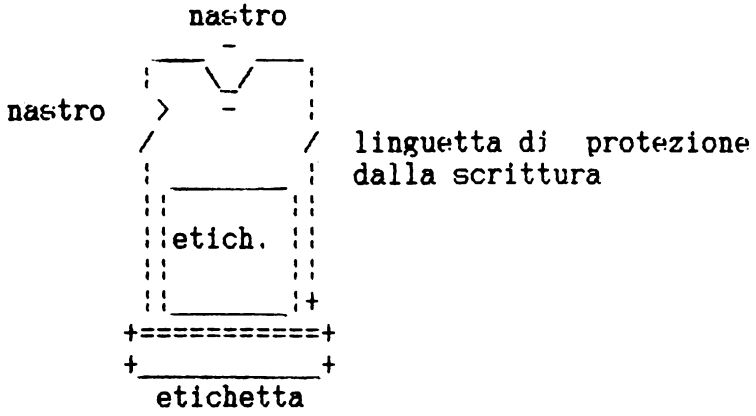
APPENDICE A - MICRODRIVE

Si consiglia di inizializzare le cartucce vergini per Microdrive piu' volte, cio' favorisce il condizionamento del nastro consentendo di conseguenza una migliore capacita' di registrazione. Per inizializzare una cartuccia si deve usare il comando di FORMAT nella seguente forma:

FORMAT MDVnumero nome

dove il numero e' quello del Microdrive interessato, mentre il nome e' quello che si intende assegnare alla cartuccia. Si faccia attenzione al fatto che l'operazione di FORMAT cancella tutte le informazioni presenti sul nastro.

Ed ecco le parti principali di una cartuccia:



Dal momento che, almeno per ora, i Microdrive sono il principale sistema di registrazione dei dati del QL, vanno trattati con cura.

E' buona norma effettuare almeno una copia del materiale importante, per salvaguardarsi se qualche cosa va male con

la cartuccia che si sta usando. Poiche' ogni cartuccia puo' contenere circa 100K byte, sara' sufficiente disporre di un paio di cartucce per le copie di sicurezza. Prima che le cartucce delle copie siano piene occorrera' un po' di tempo.

Il nastro contenuto in una cartuccia e' lungo circa 5 metri ed e' fatto girare all'interno della piccola scatola di plastica ad una velocita' di circa 2,6 km/h. Questa velocita' puo' sembrare non tanto elevata, si consideri pero' che le testine di lettura scorrono piu' di 30.000 metri di nastro all'ora.

Sebbene il nastro contenuto nelle cartucce sia un nastro video, quindi di ottima qualita', tuttavia non e' eterno ne' invulnerabile, quindi va maneggiato con attenzione. Se il nastro dovesse sporgere in fuori (nei punti dove e' indicato in figura), deve essere rimandato indietro nella cartuccia con delicatezza senza toccarlo assolutamente con le dita.

Le informazioni registrate sulle cartucce possono venire danneggiate da continue ed inutili inserzioni ed estrazioni della cartuccia oppure da accensioni e spegnimenti del QL con le cartucce inserite nei Microdrive.

Sempre si deve tener presente che le minuscole cartucce sono delle "fragili bestioline" per cui hanno bisogno di tante cure.

Al QL possono essere collegati al piu' altri sei Microdrive. Il comando DIR (usato nella forma DIR MDV1_ oppure DIR "MDV_2") serve a fornire lo spazio disponibile e l'elenco dei file della cartuccia nel Microdrive specificato.

Il comando di COPY viene utilizzato per copiare le informazioni da un dispositivo all'altro del sistema. Il canale sorgente (dispositivo) viene collegato alla destinazione con la parola TO, come in:

```
COPY MDVsorgente_nomefile TO MDVdestinazione_nomefile
```

Se si vuole cancellare un particolare file da una cartuccia,

basta usare il comando DELETE nella seguente forma:

```
DELETE MDVnumero_nomefile
```

Per caricare un programma da un Microdrive si usa il comando LOAD MDVnumero_nomefile, con questo comando viene effettuato automaticamente un NEW e viene controllato il caricamento. Se si vuole evitare il NEW si deve usare MERGE al posto di LOAD. Se si desidera che il programma vada automaticamente in esecuzione dopo il caricamento, come con il comando di CHAIN di alcuni computer quali il BBC Micro, si usi LRUN invece di LOAD e MRUN invece di MERGE.

Procedure, comandi e funzioni di controllo dei programmi:

```
AUTO inizio,passo
CLEAR
CONTINUE
EXEC disp_nomefile,...
EXEC_N disp_nomefile,...
LBYTES disp_nomefile,ind.inizio
LIST [#canale,] inizio TO fine,...
LOAD disp_nomefile
LRUN disp_nomefile
MERGE disp_nomefile
MRUN disp_nomefile
NEW
RENUM inizioprec  TO fineprec;inizio,passo
RETRY
RUN inizio
SAVE disp_nomefile inizio TO fine,...
SBYTES disp_nomefile ind.inizio,lunghezza
SEXEC disp_nomefile,ind.inizio,lunghezza,spaziodati
STOP
```

Miscellanea di altri comandi e funzioni:

```
CHR$(cod.carattere)        PEEK_W(indirizzo)
CODE(stringa)              POKE indirizzo,byte
MODE 4/8/256/512           POKE_L indirizzo,parolalunga
PEEK (indirizzo)           POKE_W indirizzo,parola
PEEK_L(indirizzo)          RND minimo TO massimo
```

APPENDICE B - FUNZIONI MATEMATICHE

Il QL dispone di un notevole numero di funzioni trigonometriche e matematiche piu' alcune di conversione. L'elenco che segue e' una raccolta di queste funzioni con alcuni dettagli. Si noti che gli angoli per le funzioni trigonometriche (ad eccezione di ACOT e ATAN) devono essere espressi in radianti.

PI -non e' una vera e propria funzione in quanto fornisce la costante pi-greco ed e' senza argomento. ad esempio PRINT PI da' il valore 3.141593

COS -fornisce il coseno di un angolo compreso fra -60000 e +60000. Si ricorda che l'argomento di una funzione deve essere racchiuso fra parentesi, come in COS(564).

SIN -da' il valore del seno dell'argomento. I limiti sono come quelli di COS fra -60000 e +60000. Esempio SIN(PI/4).

TAN -restituisce il valore della tangente trigonometrica dell'argomento, che deve essere compreso fra -30000 e +30000. Esempio TAN(3.142).

COT -produce il valore della cotangente dell'argomento, i cui limiti sono -30000 e +30000 come per TAN.

ATAN-determina l'arcotangente dell'argomento, ovvero calcola in radianti l'arco (o l'angolo) la cui tangente e' l'argomento. Diversamente dalle funzioni sopra indicate non vi sono limiti per l'argomento (salvo quelli generali del QL). E' usata nella solita forma ATAN(argomento).

ATAN-determina l'arcocotangente dell'argomento. E' del tutto analoga alla precedente, basta sostituire la parola cotangente alla parola tangente.

DEG -e' la prima delle funzioni di conversione, fornisce la

misura in gradi di un argomento in radianti. Ad esempio DEG(PI) da' 180.

RAD -e' l'inversa della precedente e produce il numero di radianti corrispondente a quello dell'argomento in gradi. Ad esempio RAD(180) da' 3.141593.

EXP -e' la funzione esponenziale e con essa si ottiene il risultato di e elevato all'argomento. Affinche' il risultato non superi i limiti aritmetici del QL (overflow) l'argomento deve essere compreso fra -500 e +500. la sua forma e' al solito EXP(argomento).

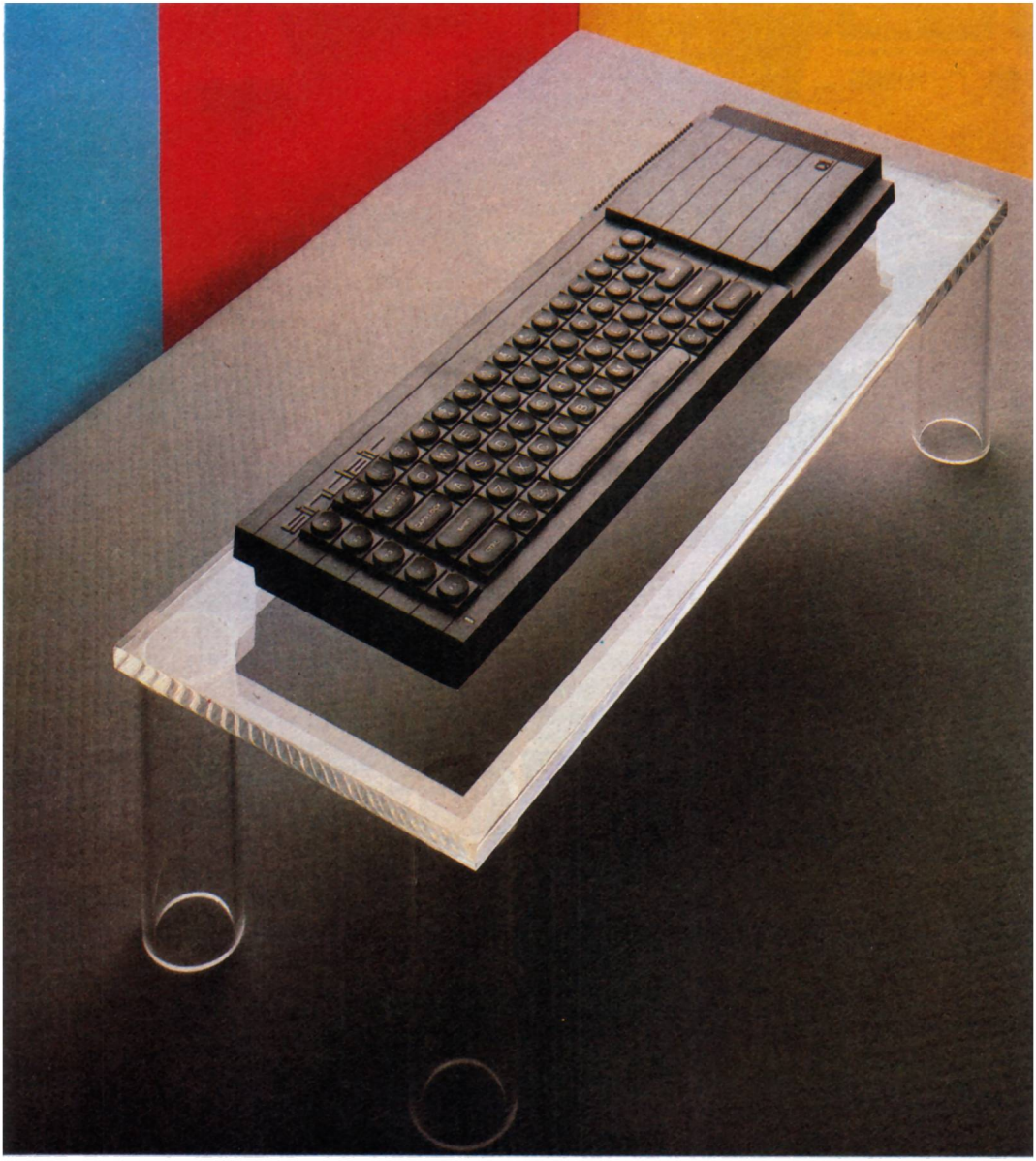
LN -serve a calcolare il logaritmo naturale (in base e) dell'argomento, e' l'inversa di EXP. L'argomento deve essere maggiore di zero e non vi sono limiti superiori (salvo quelli del QL).

LOG10 simile alla precedente, unica differenza e' la base che in questo caso e' 10, cioe' il risultato e' logaritmo decimale dell'argomento, con le stesse limitazioni di LN.

INT -fornisce la parte intera dell'argomento. Viene cioe' effettato il troncamento delle cifre decimali. L'argomento deve essere compreso fra -32767 e +32767. Ad esempio INT(29.8765) da' 29 e INT(-28.789) da' -28.

ABS restituisce il valore assoluto dell'argomento. Percio' si ottiene l'argomento stesso se positivo, il suo opposto se negativo. Ad esempio ABS(674)=674 mentre ABS(-3452)=3452.

SQRT-produce la radice quadrata dell'argomento, che non deve essere negativo. L'argomento anche qui deve essere messo fra parentesi tonde. Ad esempio SQRT(876).



L'ultimo nato della Sinclair si scosta dalla linea estetica tradizionale per il suo basso profilo e per la forma dei tasti dalla superficie concava.

APPENDICE C - SOFTWARE PROFESSIONALE

Il software applicativo fornito (gratuitamente!) con il Sinclair QL e' stato scritto dalla Psion Ltd e *obiettivamente si puo' sostenere che esso si pone sui piu'* alti livelli del software esistente per i microcomputer.

Solo la potenza del QL puo' supportare le capacita' e la semplicita' d'uso del QL ABACUS, ARCHIVE, EASEL e QUILL, rispettivamente un tabellone elettronico, un database, un pacchetto per la rappresentazione grafica ed uno per l'elaborazione dei testi.

Questi programmi sono stati appositamente studiati e scritti per il QL e comprendono soluzioni cosi' avanzate da renderli superiori al software dello stesso genere attualmente in circolazione.

Tutti i programmi sono completamente interattivi e i risultati delle elaborazioni possono essere visualizzati, stampati o memorizzati istantaneamente con dei semplici comandi.

Una serie di utili note e promemoria e' costantemente visualizzata sullo schermo e puo' essere eliminata o resa visibile premendo il tasto di funzione F2, il quale come alcuni altri tasti (indicati nel seguito) e' comune ai quattro pacchetti applicativi.

Quindi non e' richiesto di imparare a memoria o di ricercare sul manuale lunghi elenchi di istruzioni e comandi. Anzi, poiche' questo software e' stato progettato come un insieme unico ed integrato, i programmi sono omogenei al punto che basta studiarne a fondo uno per conoscerli tutti.

Tutti impiegano i colori per una migliore visualizzazione e tutti possono scambiarsi fra loro i dati e le informazioni. Ad esempio, i valori provenienti dall'elaborazione del tabellone elettronico possono essere trasferiti a QL EASEL per una loro immediata rappresentazione grafica.

In tutte e quattro le applicazioni premendo il tasto di funzione F1, in qulsiasi momento, si ottengono spiegazioni su cio' che si sta facendo e un menu di comandi disponibili in quel momento con le relative modalita' d'uso. In sostanza F1 e' il tasto di richiesta di AIUTO.

Altro tasto comune e' F3, che serve per accedere ai comandi disponibili in quel momento, mentre si deve usare ESC per uscire da una situazione qualsiasi, ad esempio per porre termine alle spiegazioni oppure per ripristinare l'esecuzione dopo un comando.

Tutti i programmi possono operare indifferentemente in tre modi grafici diversi cioe' con schermo a 40/64/80 colonne. Cio' dipende essenzialmente dal tipo di display che si ha a disposizione.

Se si vuole utilizzare la modalita' a 80 colonne e' necessario dotarsi di un monitor, data l'elevata definizione che un comune televisore non possiede. La modalita' a 64 colonne e' soddisfacente su un monitor, ovviamente, e su televisori di buona marca, mentre quella a 40 e' riservata agli ... altri.

Ogni programma ha una lunghezza variabile fra i 60K e i 90K byte con una struttura overlay, cioe' con occupazione della memoria solo della parte di programma che implementa i comandi piu' frequenti. Le altre parti vengono richiamate in memoria solo se sono richiesti particolari comandi non di uso frequente.

Le cartucce con i programmi PSION (o meglio le loro copie) devono sempre occupare il Microdrive 1 (quello di sinistra), mentre quelle destinate a contenere dati ed informazioni vanno inserite nel Microdrive 2. Salvo alcune eccezioni, indicate sul manuale, le cartucce non devono mai essere tolte fino al termine dell'utilizzo dei programmi.

QL ABACUS

Si tratta di uno dei piu' popolari programmi per mini e microcomputer e tradizionalmente uno dei piu' difficili da dominare. In questo caso tutto e' risolto, ogni difficolta' e' stata appianata.

QL ABACUS esegue calcoli e simulazioni in tempo reale con precisione e rapidita'. Facilmente consultabili sul manuale, sono forniti esempi di applicazioni quali l'analisi dei movimenti di cassa, la gestione del conto corrente bancario, la gestione del preventivo delle entrate/uscite, i piani di ammortamento, l'analisi di Fourier e altro. Le applicazioni, come si puo' constatare, sono innumerevoli ed hanno come unico limite la fantasia.

A differenza di altri spreadsheet, QL ABACUS consente il riferimento a righe colonne e celle con un nome scelto a piacere, in tal modo si puo' evitare di usare lettere e numeri privi di significato.

E' possibile, inoltre, assegnare ad un tasto funzione il compito di modificare il valore di una variabile e di eseguire un calcolo di simulazione completa, in tal modo premendo un solo tasto si realizza la situazione di "What if" (cosa accadrebbe se ...).

QL ARCHIVE

QL ARCHIVE e' un sistema di archiviazione intelligente dei dati. Cioe' con esso si puo' memorizzare qualsiasi genere di informazione, scegliendo sia le modalita' di registrazione che quelle di ricerca e rappresentazione, in altre parole e' un database.

Questo del QL ARCHIVE e' del tipo relazionale completo e multifile; cio' significa che puo' operare su piu' file contemporaneamente utilizzando le relazioni fra i campi dei record di ogni file.

QL ARCHIVE e' addirittura fornito di un suo proprio linguaggio di programmazione, simile al SuperBASIC (dove pero' le linee non devono essere numerate), con il quale si possono scrivere delle procedure che possono arricchire di nuovi comandi tutta l'applicazione.

La gestione dei record e' del tutto trasparente all'utente, che non deve preoccuparsi nemmeno di conoscere prima le loro dimensioni.

I comandi includono opzioni di semplice e rapida ricerca, ordinamento e selezione dei record, numerosi operatori per la manipolazione delle stringhe e una veloce ed accurata aritmetica.

Nel manuale non mancano numerosi esempi di applicazioni ne' un certo numero di procedure di interesse generale che possono benissimo essere inserite nei propri lavori. In piu' nella cartuccia e' presente un esempio completo di file contenente dei dati sui paesi del mondo.

A titolo dimostrativo si riporta l'elenco delle parole chiave e delle funzioni del linguaggio di QL ARCHIVE, tutte scritte in lettere minuscole solo per distinguerle da quelle del SupeBASIC.

PAROLE-CHIAVE di QL ARCHIVE

all	find	lprint	screen
endall	first	merge	search
alter	format	mode	sedit
append	if	new	select
back	else	next	sinput
backup...as	endif	open	sload
close	import...as	order	spooloff
cls	ink	paper	spoolon
continue	input at;tab;	position	sprint
create	insert	print	ssave
endcreate	kill	proc	stop
delete	last	endproc	trace
dir	let	quit	update
display	llist	rem	use
dump	load	reset	while
edit	local	return	endwhile
error	locate	run	
export	look	save	

FUNZIONI di QL ARCHIVE

abs(numero)	come ABS	instr(str,substr)	come INSTR
chr(numero)	come CHR$	int(numero)	come INT
code(stringa)	come CODE	lower(stringa)	rende minusc
count(file)	nr.record file	memory()	byte liberi
date(0/1/2)	forma data	month(numero)	mese in lett
days(stringa)	nr.gg 1/1/1583	numfld(file)	nr. campi
eof(file)	fine file	recnum(file)	nr. record
errnum()	nr. errore	rept(str,numero)	str. X nr
fieldt(n,fil)	tipo campo	sgn(numero)	segno
fieldv(n,fil)	valore campo	str(n,tipo,dec)	forma numero
found()	record trovato	sqr(numero)	rad.quadrata
getkey()	come INKEY$(-1)	time()	hh:mm:ss
inkey()	come INKEY$	upper(stringa)	rende maiusc

QL EASEL

L'unicita' di questo pacchetto per la rappresentazione grafica dei dati e' veramente esemplare e tale da renderlo non confrontabile con alcun programma del genere in commercio. La compatibilita' con QL ABACUS e' totale ed i file provenienti dal tabellone elettronico possono essere istantaneamente visualizzati graficamente da QL EASEL.

I dati possono essere elaborati e rappresentati in ben otto differenti versioni di base che possono poi essere ampliate e variate a piacere. Grafici a barre orizzontali e verticali, grafici a torta, grafici lineari sono alla portata di tutti in una varieta' di soluzioni e di accostamenti di colore senza fine.

Ma QL EASEL e' piu' di uno strumento di rappresentazione grafica di dati: e' un tabellone grafico-elettronico. E' infatti possibile inserire direttamente i dati, stabilire delle relazioni fra loro e vedere istantaneamente in forma grafica i risultati.

lenza sostituirsi alla potenza di QL ABACUS, QL EASEL e' il primo tabelone grafico-elettronico disponibile. I testi possono essere inseriti in tutte le direzioni senza problemi, e con altrettanta semplicita' i grafici si possono ottenere in copia stampata (se la stampante e' grafica) o memorizzati come file sui nastri per Microdrive.

QL QUILL

Nell'ambito dell'elaborazione dei testi (word processing) il QL QUILL stabilisce certamente un nuovo standard di perfezione. Esso impiega tutta la potenza del QL per mettere a disposizione dell'utente un impressionante numero di comandi, opzioni e facilitazioni.

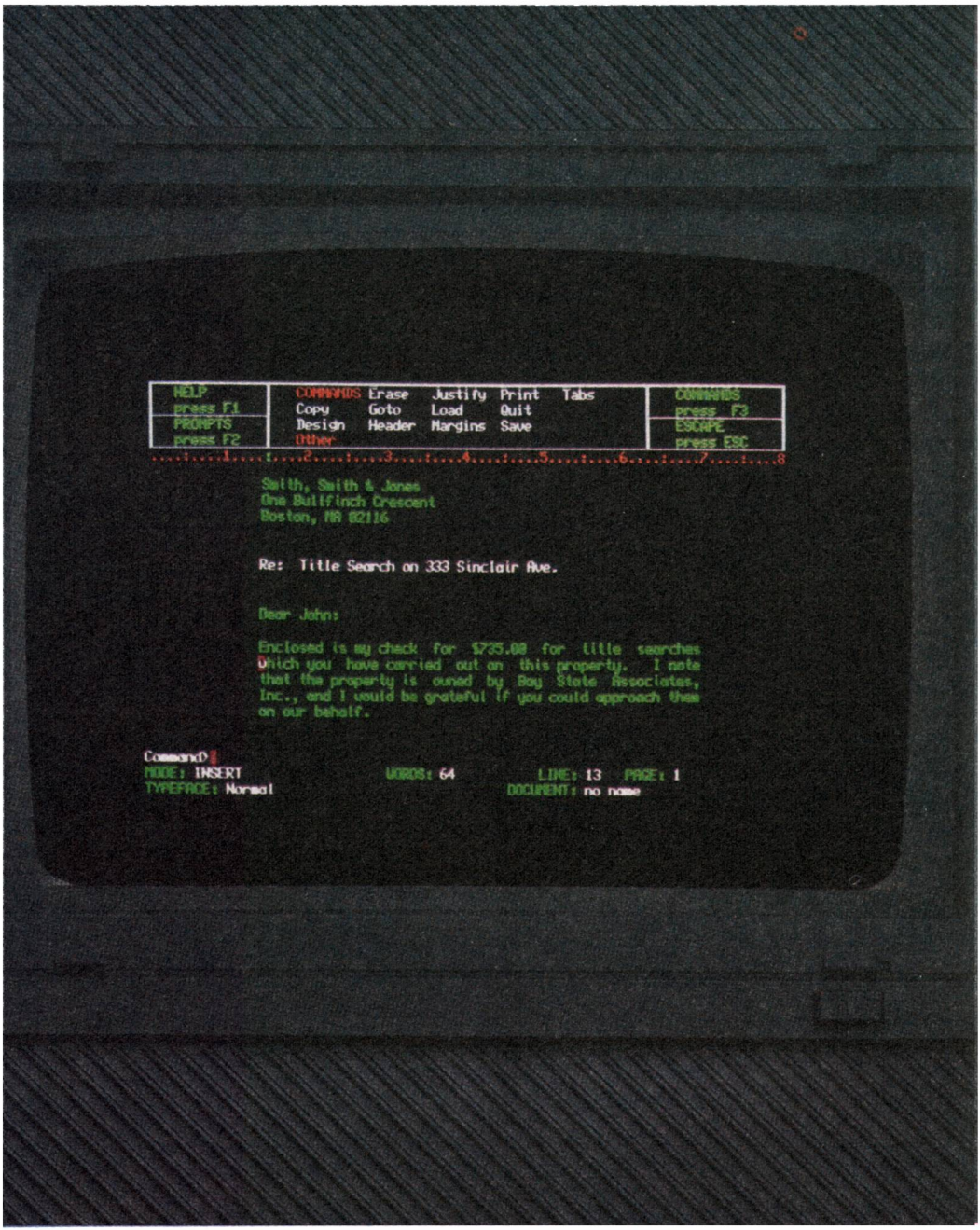
In esso il dispaly e' a piu' colori (modificabili) per la massima definizione dei dettagli , e nemmeno il dattilografo piu' veloce e preciso puo' competere con la sua velocita' e le sue possibilita' di editing.

QL QUILL e' un pacchetto di programmi per l'elaborazione dei testi talmente avanzato e perfezionato che potrebbe giustificare da solo l'acquisto del Sinclair QL.

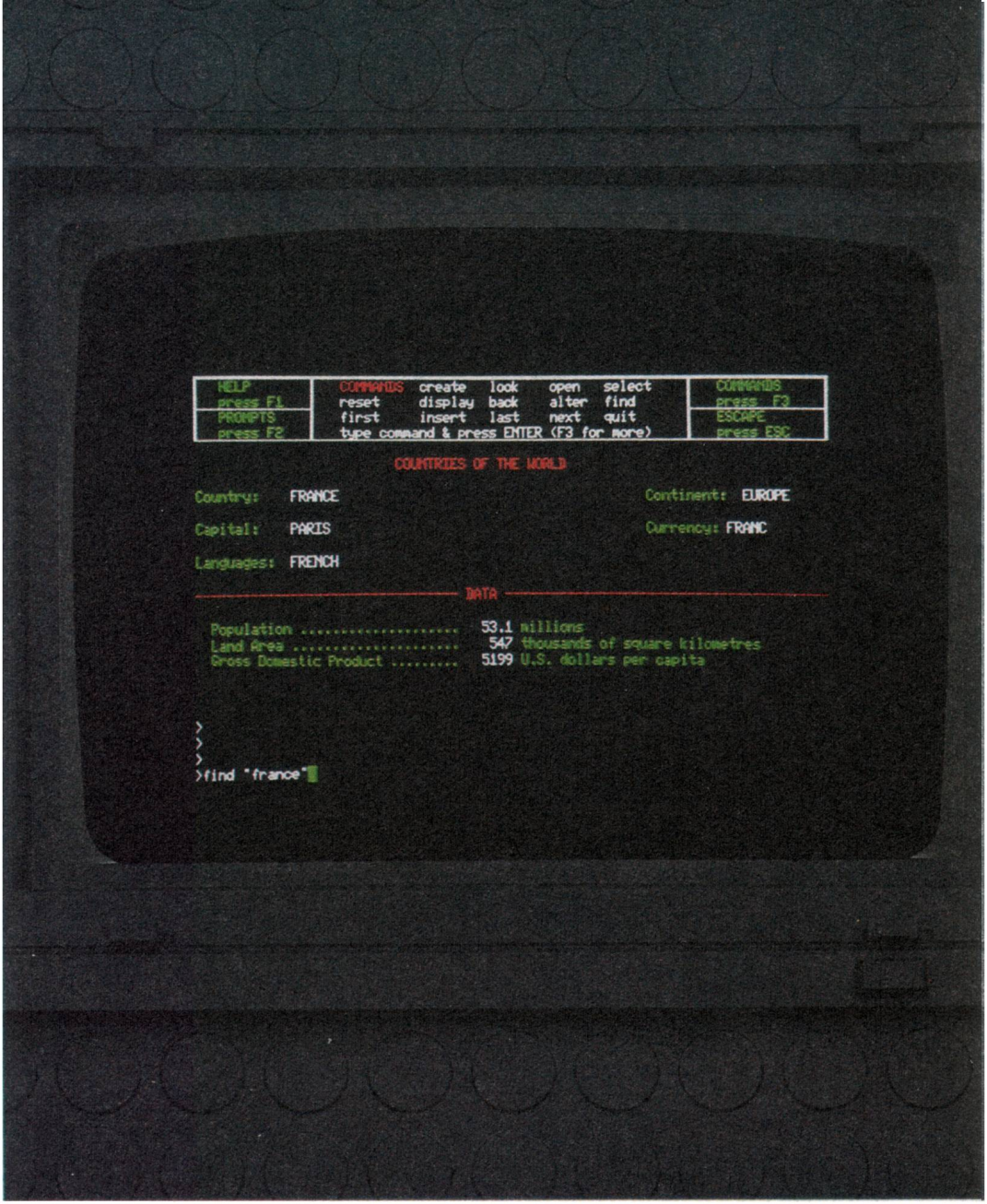
Per imparare ad usarlo bastano pochi minuti, esso infatti e' dotato di un superbo tasto di AIUTO (F1) che fornisce dettagliate spiegazioni di ogni cosa in qualsiasi momento.

Anche qui a titolo di curiosita' si riporta l'elenco dei principali comandi (ottenibili premendo solo il tasto della loro prima lettera dopo F3), ognuno dei quali presenta diverse opzioni:

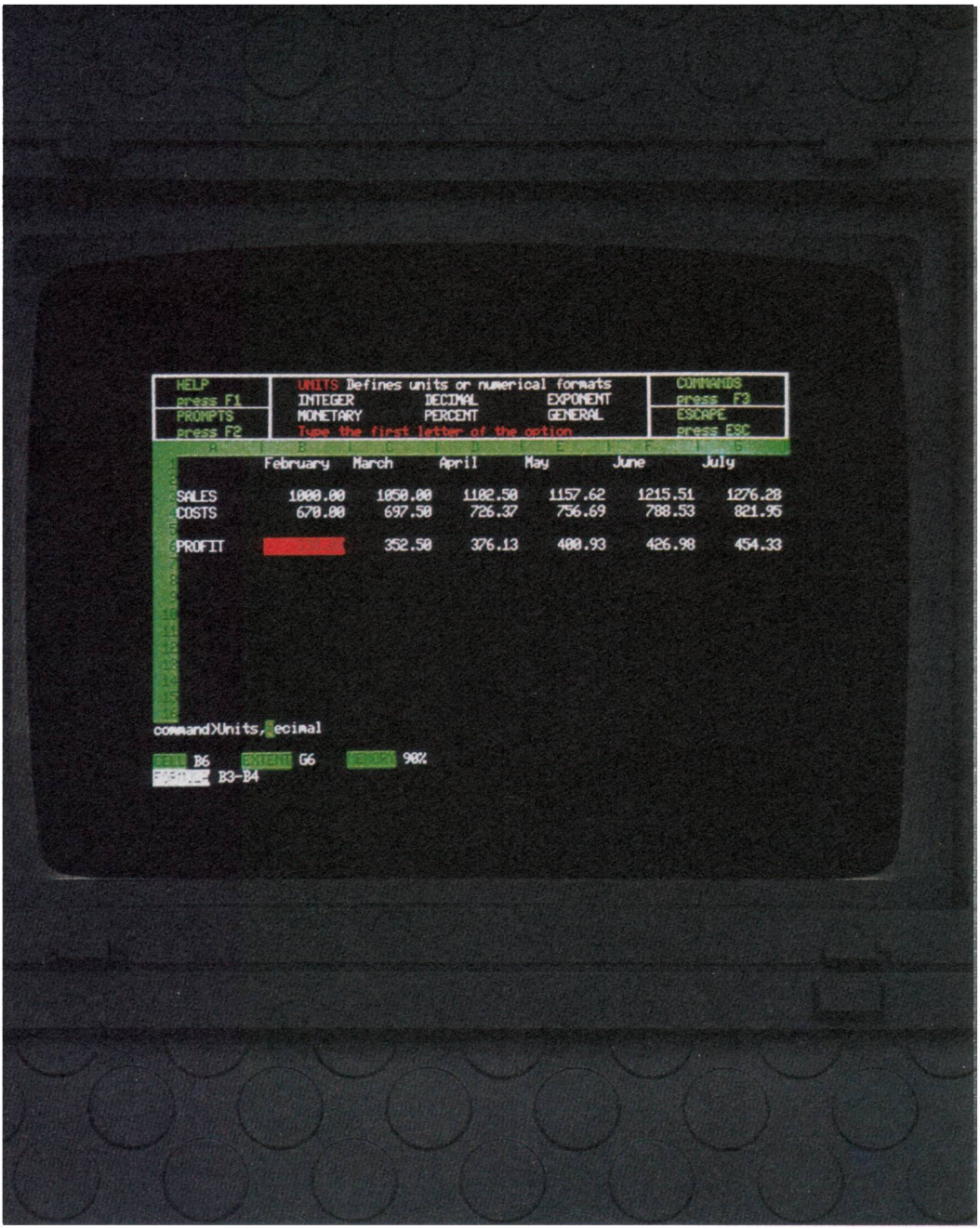
Copy	Margins
Design	Merge
Erase	Page
Files	Print
Footer	Quit
Goto	Replace
Header	Save
Hyphenate	Search
Justify	Tabs
Load	Zap



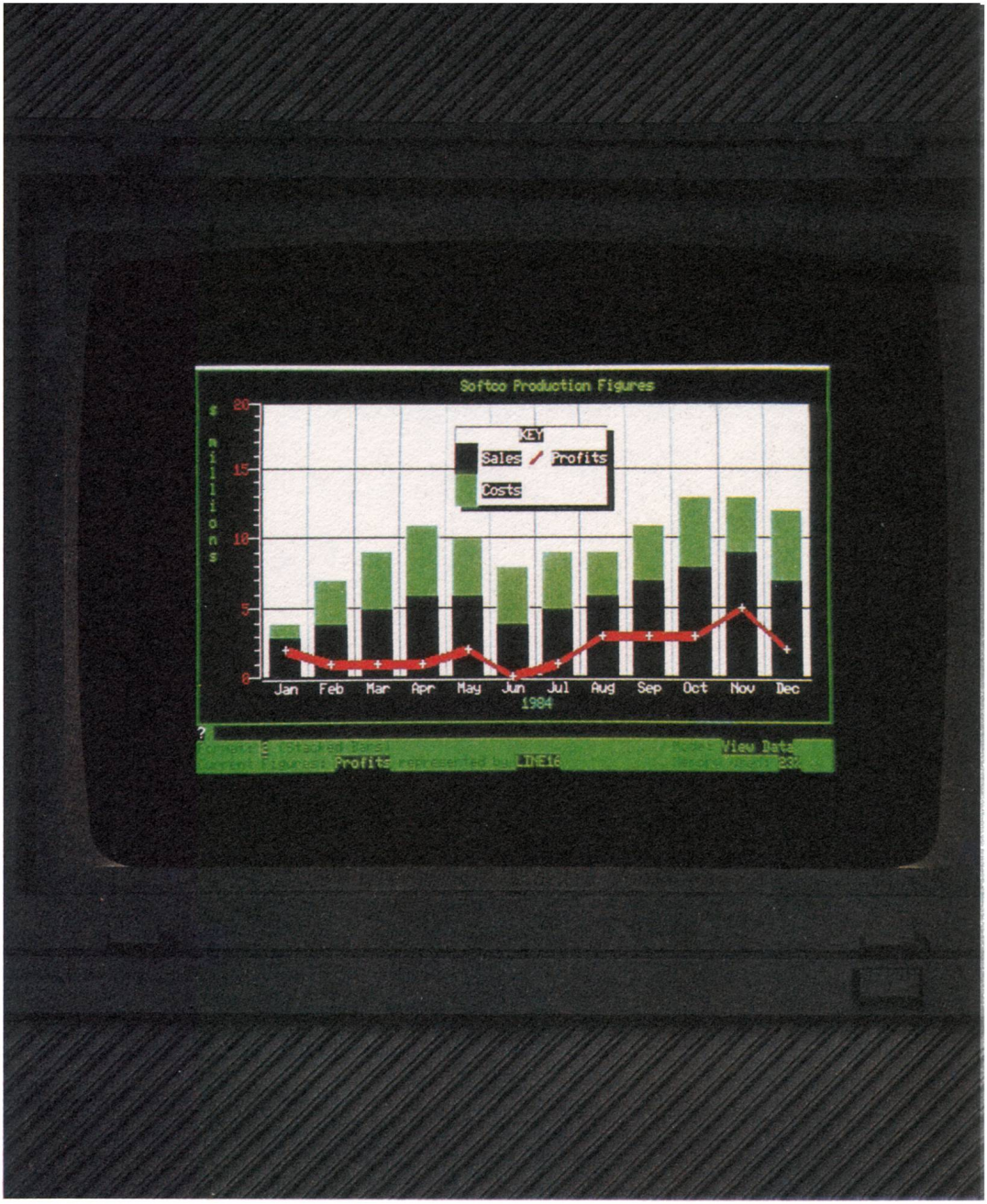
Il Word Processor è uno dei quattro programmi interattivi scritti appositamente per il QL. Ecco come si presenta.



Questa videata mostra il programma di Database il quale può mettere a disposizione un libro indirizzi di 256 campi.



L'Abacus è un vero e proprio catalogo da compilare per mezzo delle coordinate della riga e della colonna.



Il QL possiede enormi possibilità grafiche in quanto può creare e mescolare barre e linee diversamente colorate.

APPENDICE D - PAROLE CHIAVE DEL SUPERBASIC

ABS	funzione matematica	ACOT	funzione matematica
ADATE	orologio	ARC	grafica
ARC_R	grafica	AT	finestre
ATAN	funzione matematica	AUTO	basic
BAUD	comunicazioni	BEEP	suono
BEEPING	suono	BLOCK	finestre
BORDER	finestre	CALL	linguaggio macchina
CHR$	basic	CIRCLE	grafica
CIRCLE_R	grafica	CLEAR	basic
CLOSE	dispositivi	CLS	finestre
CODE	basic	CONTINUE	basic
COPY	dispositivi	COPY_N	dispositivi
COS	funzione matematica	COT	funzione matematica
CSIZE	schermo	CURSOR	finestre
DAY$	orologio	DATA	basic
DATE	orologio	DATE$	orologio
DEG	funzione matematica	DELETE	microdrive
DIM	vettori e schiere	DIMN	vettori e schiere
DIR	dispositivi	DIV	operatore
DLINE	basic	EDIT	basic
EXEC	multitasking	EXEC_N	multitasking
EXIT	cicli	EXP	funzione matematica
FILL	grafica	FILL$	stringhe
FLASH	finestre	FORMAT	microdrive
GOSUB	basic	GOTO	basic
INK	finestre	INKEY$	basic
INPUT	finestre	INSTR	stringhe
INT	funzione matematica	KEYROW	tastiera
LBYTES	dispositivi	LEN	stringhe
LET	basic	LINE	grafica
LINE_R	grafica	LIST	basic
LN	funzione matematica	LOAD	dispositivi
LOCAL	funzioni/procedure	LOG10	funzione matematica
LRUN	basic	MERGE	dispositivi
MOD	operatore	MODE	schermo
MOVE	grafica	MRUN	basic
NET	dispositivi	NEW	basic

NEXT	cicli	ON..GOSUB	basic
ON..GOTO	basic	OPEN	dispositivi
OPEN_IN	dispositivi	OPEN_NEW	dispositivi
OVER	finestre	PAN	finestre
PAPER	finestre	PAUSE	multitasking
PEEK	basic	PEEK_W	basic
PEEK_L	basic	PENUP	grafica
PENDOWN	grafica	PI	funzione matematica
POINT	grafica	POINT_R	grafica
POKE	basic	POKE_W	basic
POKE_L	basic	PRINT	basic
RAD	funzione matematica	RANDOMIZE	numeri casuali
READ	basic	RECOL	schermo
REMark	basic	RENUM	basic
RESTORE	basic	RETRY	basic
RETurn	funzioni/procedure	RND	numeri casuali
RUN	basic	SAVE	dispositivi
SBYTES	dispositivi	SCALE	finestre
SCROLL	finestre	SEXEC	multitasking
SDATE	orologio	SIN	funzione matematica
SQRT	funzione matematica	STOP	basic
STRIP	finestre	TAN	funzione matematica
TURN	grafica	TURNTO	grafica
UNDER	finestre	WINDOW	finwstre

DEFine FuNction/END DEFine	funzioni e procedure
DEFine PROCedure/END DEFine	funzioni e procedure
FOR/END FOR	cicli
IF...THEN...ELSE/END IF	condizioni
REPeat/END REPeat	cicli
SELect...ON...REMAINDER/END SELect	condizioni
TO	operatore

Progettato per una migliore e più lineare realizzazione dei programmi, il Super BASIC SINCLAIR è quanto di più avanzato si possa immaginare nel campo dei linguaggi di programmazione, al passo con le ultime innovazioni hardware.
La creazione di procedure, la programmazione strutturata, la grafica sofisticata ad altissima risoluzione sono solo alcune delle innumerevoli risorse di questo linguaggio.
ANDREW NELSON, autore di molti libri sui calcolatori, spiega i concetti fondamentali, le parole-chiave, i comandi e le istruzioni del Super BASIC.
In questo libro l'autore mostra come tutte le capacità acquisite col BASIC dello SPECTRUM siano direttamente trasferibili sul QL, ed illustra la facilità con la quale si possono rendere più raffinati e potenti i programmi, dopo aver appreso alcune delle nuove istruzioni e dei nuovi comandi del ricchissimo vocabolario del Super BASIC.
La presente versione italiana del libro, a cura di Severino Grandi, laureato in matematica e appassionato conoscitore dello SPECTRUM, contiene in più rispetto all'edizione originale, la presentazione dei quattro programmi applicativi forniti insieme al QL. Essi costituiscono un corredo talmente formidabile e ricco al punto di consentire anche ai meno provveduti, in fatto di calcolatori, di accedere immediatamente alla tastiera del QL.

Cod. 9050

L. 20.000

ALLA SCOPERTA DEL QL IL COMPUTER SINCLAIR di ANDREW NELSON

edizioni JCE